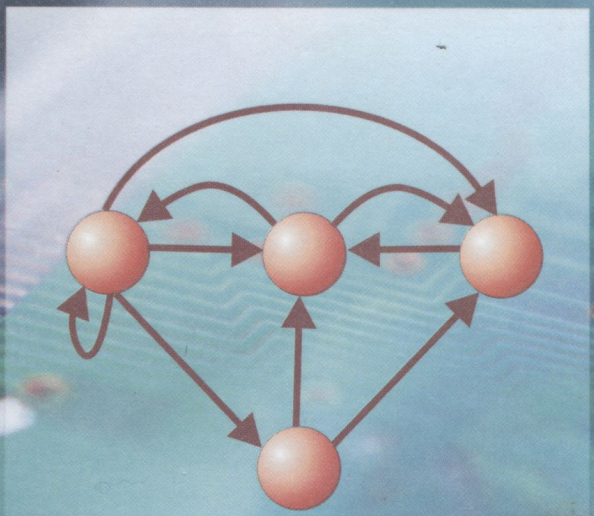
С. Л. Кривий

КУРС ДИСКРЕТНОЇ МАТЕМАТИКИ

Навчальний посібник

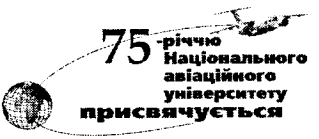


$$(\neg B \rightarrow \neg A) \leftrightarrow (A \rightarrow B)$$

С. Л. Кривий

КУРС ДИСКРЕТНОЇ МАТЕМАТИКИ

Рекомендовано
Міністерством освіти і науки України
як навчальний посібник для студентів
вищих навчальних закладів



Київ
Книжкове видавництво
Національного авіаційного університету
2007

УДК 51.681
ББК В176я7
К 821

Рецензенти:

А. О. Чикрій, д-р фіз.-мат. наук, проф., чл.-кор. НАН України
(Інститут кібернетики НАН України)

М. П. Моклячук, д-р фіз.-мат. наук, проф.
(Київський національний університет імені Тараса Шевченка)

А. Ю. Дорошенко, д-р фіз.-мат. наук, проф.
(Київський національний технічний університет «КПІ»)

Гриф надано Міністерством освіти і науки України
(Лист № 1.4/18-Г-667 від 07.08.06)

Кривий С. Л.

К 821 Курс дискретної математики: Навч. посібник. — К.: Книжкове вид-во НАУ, 2007. — 432 с.

ISBN 966–598–353-9

До навчального посібника ввійшли розділи, які використовують основні поняття теорії множин та відношень, загальної алгебри, математичної логіки і теорії алгоритмів. Наступний розділ присвячений основним алгоритмічним системам — частково рекурсивним функціям, машинам Поста, машинам Тьюрінга і алгоритмам Маркова. Ці системи застосовуються для уточнення поняття алгоритму, означення алгоритмічної розв'язуваності та встановлення алгоритмічної повноти мов програмування. Розглянуто основні поняття та методи теорії графів.

Представлено формальні логічні мови, такі як логіка висловлювань, лінійна темпоральна логіка та логіка предикатів першого порядку.

Одним з найбільших розділів даного посібника є розділ з теорії складності обчислень за Тьюрінгом. Машини Тьюрінга, як одна з найбільш уживаних алгоритмічних систем, служать еталоном виміру складності обчислень, що дає змогу якісно оцінювати алгоритми розв'язку однієї і тієї самої задачі. У цьому розділі розглянуто найважливіші модифікації машин Тьюрінга. До цих модифікацій, зокрема, відносяться односторонні, недетерміновані та багатострічкові машини Тьюрінга, а також описані такі моделі обчислень, як RAM і PRAM (для оцінки послідовних та паралельних алгоритмів).

Для студентів вищих навчальних закладів та аспірантів, які навчаються за напрямом «Комп'ютерні науки».

УДК 51.681
ББК В176я7

ISBN 966–598–353-9

ЗМІСТ

ПЕРЕДМОВА

Нині ми є свідками того, як змінюється співвідношення дискретної математики і класичної (неперервної) математики. Протягом останніх двох з половиною століть основну роль у вивченні явищ природи відігравав математичний аналіз — диференціальне та інтегральне числення, диференціальні рівняння математичної фізики, варіаційне числення, методи функцій комплексної змінної тощо. Процеси, що мали атомістичну природу, змінювалися неперервними з метою застосування до їх дослідження добре розвинутого апарату неперервної математики. У зв'язку з цим дискретна математика тривалий час була на правах Попелюшки, краса якої затьмарювалася блиском впливових і сильних сестер.

Стан справ докорінно змінився після того, як з'явилися швидкодіючі електронні обчислювальні машини (ЕОМ). З появою ЕОМ такі абстрактні області математики, як математична логіка, загальна алгебра, комбінаторика, формальні граматики і т. п., стали прикладними областями. Важливу роль почали відігравати дискретні моделі, які виникли в задачах економічного характеру, задачах проектування технічних пристроїв, у задачах побудови літальних та космічних апаратів. Необхідність розв'язування цих і їм подібних задач приводять до необхідності підготовки спеціалістів відповідного профілю. Пропонований навчальний посібник саме і має на меті служити цій справі.

Цей посібник написано відповідно до програми курсу «Основи дискретної математики», він розрахований на 3—4-семестровий відрізок часу. Джерелами інформації для цього посібника послужили деякі матеріали підручника «Основи дискретної математики» (К.: Наукова думка,

автори Капітонова Ю. В., Кривий С. Л., Летічевський О. А., Луцький Г. М., Печурін М. К., 2002) та матеріали журнальних статей і монографій, на які є відповідні посилання. Матеріали розділів, які перекликаються з розділами вищеназваного підручника, розширені як додатковими відомостями, так і новими задачами. Цілком новими є розділи «Елементи теорії складності обчислень», «Зображення булевих функцій» та «Лінійна пропозиційна темпоральна логіка». Ці розділи з'явилися у зв'язку з певними критичними зауваженнями, що надійшли на адресу автора (як одного із авторів вищеназваного підручника), а також у зв'язку з появою нових методів верифікації дискретних динамічних систем. У першому з названих розділів описано основні поняття теорії складності обчислень за Тьюрінгом, наведено основні класи складності та поліноміальна ієрархія класів складності алгоритмів. Представлено також моделі для оцінки складності паралельних алгоритмів і програм. У другому з названих розділі описано лінійну темпоральну логіку і метод семантичного табло для перевірки виконуваності формул цієї логіки. Мова темпоральної логіки в даний час широко використовується для представлення конкретних властивостей дискретних динамічних систем.

Автор ставив за мету систематичне викладення засобів дискретної математики як інструментарію для розв'язання задач, які виникають під час проектування та верифікації, передусім програмного і технічного забезпечення ЕОМ.

Матеріал подається на основі аксіоматичного підходу і тому від читача вимагається певна математична підготовка і мінімальні знання з вищої алгебри та математичного аналізу. Матеріал викладено максимально замкнуто.

Посібник розрахований на студентів та аспірантів технічних вузів, які навчаються за напрямом «Комп'ютерні науки», і може служити основою для таких спецкурсів, як бази даних і бази знань, теорія алгоритмів, комп'ютерна алгебра, методи верифікації програм, криптографічні методи захисту інформації тощо.

Автор

1 МНОЖИНИ І ВІДНОШЕННЯ

Поняття множини — одне з основних, якщо не основне, поняття математики. Воно не має точного означення, і його відносять до аксіоматичних понять. Такими аксіоматичними поняттями, наприклад, в елементарній геометрії є поняття «точка», «пряма», «площина».

Як правило, термін **множина** пояснюється за допомогою прикладів, а потім вказуються правила його використання в математичних застосуваннях. Останнє можна зробити на різних рівнях строгості. Детальне і строге викладення теорії множин вимагало б скрупульозного аналізу логіки математичних суджень, а це — спеціальна самостійна тема, котра відноситься до області *основ математики*. Для наших цілей достатньо рівня так званої **інтуїтивної теорії множин**.

1.1. Множини, відношення, функції, операції

Інтуїтивне поняття множини. Основні принципи. Як зазначалося вище, поняття множини відноситься до аксіоматичних понять математики і точне його означення дати неможливо. Часто приймається формулювання інтуїтивного поняття множини Г. Кантора, основоположника цієї теорії.

Довільне зібрання певних предметів нашої інтуїції чи інтелекту, які можна відрізнити один від одного і які уявляються як єдине ціле, називається множиною. Предмети, які входять до складу множини, називаються її елементами.

Суттєвим пунктом канторівського розуміння множини є те, що зібрання предметів саме розглядається як один предмет («уявляється як єдине ціле»). Основна увага тут переноситься з окремих предметів на зібрання предметів, які, у свою чергу, теж розглядаються як предмети.

Що стосується «предметів нашої інтуїції чи інтелекту», то це формулювання дає значну свободу передусім тим, що ніяк не обмежує природу предметів, які складають множину. Множина може складатися, наприклад, з людей, точок площини, простих чисел, планет Всесвіту. Зауважимо також, що канторівське формулювання дає можливість розглядати множини, елементи яких з певних причин точно визначити неможливо. У зв'язку з цим згадаємо, що елементи довільної нескінченної множини, навіть теоретично, неможливо зібрати в закінчену сукупність (з цією проблемою пов'язана філософська абстракція так званої актуальної нескінченності). Відомі також і скінченні множини, які мають таку саму міру невизначеності, як і довільна нескінченна множина.

З'ясуємо, нарешті, зміст слів «які можна відрізнити один від одного» і «певні предмети». У першому випадку для довільних двох предметів, які розглядаються як елементи даної множини, повинна існувати можливість з'ясувати, різні ці предмети чи однакові. У другому випадку, якщо задано деяку множину і який-небудь предмет, то можна визначити, є чи ні цей предмет елементом даної множини. Звідси випливає, що довільна множина повністю визначається своїми елементами. Ця канторівська вимога формулюється таким чином.

Інтуїтивний принцип об'ємності (аксіома екстенсіональності)

Дві множини рівні між собою тоді і тільки тоді, коли вони складаються з одних і тих самих елементів. Рівність двох множин *A* і *B* позначають через

$$A = B.$$

Отже, дві множини рівні, якщо кожний елемент однієї з них є елементом другої, і навпаки.

Множина називається **скінченною,** якщо вона складається із скінченного числа елементів. Запис $a \in A$ ($a \notin A$) означає, що a є (не є) елементом множини A. Та однозначно визначена множина, елементами якої є $a_1, a_2, \ldots, a_n$, позначається $\{a_1, a_2, \ldots, a_n\}$.

Приклад 1.1.1. 1. Множина $\{a\}$ — так звана *одноелементна множина* — є множиною, єдиним елементом якої є елемент a.

2. Множини $A = \{2, 4, 6\}$, $B = \{2, 6, 4\}$, $C = \{2, 2, 6, 6, 4, 4\}$, $C = \{2, 2, 4, 4, 6, 6, 6\}, \ldots$ рівні між собою, оскільки складаються з одних і тих самих елементів. Перша і друга множини відрізняються одна від одної порядком своїх елементів, а третя і четверта від пер-

ших двох тим, що в них елементи 2, 4 і 6 присутні більше ніж в одному екземплярі кожний.

3. Множини $\{\{a, b\}, \{b, c\}\}$ і $\{a, b, c\}$ не рівні між собою, оскільки перша складається з елементів $\{a, b\}$ і $\{b, c\}$, а друга — з елементів a, b, c.

4. Множини $\{\{1,2\}\}$ і $\{1,2\}$ не рівні між собою, оскільки перша одноелементна, а друга двоелементна.

Останній приклад показує, що необхідно розрізняти предмет і множину, єдиним елементом якої є цей предмет ♠[1]

Розглянемо детальніше ситуацію, яка має місце в пункті 2 попереднього прикладу. З принципу об'ємності випливає, що всі множини типу множин C, C' і т. д., які були наведені в цьому прикладі, рівні між собою і дорівнюють множині A. Введемо таке означення.

Визначення 1. *Множина, що складається з елементів деякої множини A так, що ці елементи можуть входити до складу цієї множини в якій завгодно кількості екземплярів, називається* **мультимножиною** *множини A і позначається $M(A)$.*

З погляду теорії множин, множина і її мультимножина — це один і той самий об'єкт і їх можна не розрізняти. Проте часто, особливо коли йдеться про представлення множини в пам'яті обчислювальної машини, виникає потреба відрізняти мультимножину від множини.

Задання множини за допомогою фігурних дужок з явним переліком її елементів можливе лише тоді, коли множина має невелику кількість елементів. Якщо ж множина має хоча й скінченну, але велику кількість елементів, то таке задання множини стає досить громіздким, а у випадку нескінченної множини його застосування взагалі стає неможливе. Виникає питання: як задавати множини, які мають велике або нескінченне число елементів?

Відповідь пов'язана з поняттям властивості або висловлювання. Поки що обмежимося інтуїтивним поняттям висловлювання.

Під **висловлюванням** відносно предмета x будемо розуміти таке розповідне речення, в якому щось стверджується відносно предмета x і яке можна характеризувати як істинне або хибне по відношенню до x.

Приклад 1.1.2. Властивостями є такі записи:

1) 3 ділить x; 2) $x < x$;

3) $x^2 = 2$; 4) $x^2 + 1 > 0$.

---

[1] Символ ♠ означає кінець прикладу або групи прикладів, а символ ■ — кінець доведення твердження.

Наведені нижче вирази не є властивостями.

5) Для всіх x, y $xy = yx$; 6) Існує таке x, що $2x < 0$, тому що їх не можна характеризувати як істинні чи хибні. ♠

Нехай $P(x)$ означає деяку властивість, тоді $P(a)$ буде означати ту саму властивість, але з заміною x на a. Задання множини в термінах властивостей досягається за допомогою такого принципу.

Інтуїтивний принцип абстракції (аксіома згортання)

Довільна властивість $P(x)$ визначає деяку множину A за допомогою умови: елементами множини A є ті і тільки ті предмети a, які мають властивість P.

Зважаючи на принцип об'ємності, довільна властивість $P(x)$ визначає єдину множину, яку позначають $\{a \mid P(a)\}$ і читають так: «множина всіх тих предметів a, що $P(a)$». Зауважимо, що властивість P може являти собою спосіб побудови елементів множини $\{a \mid P(a)\}$.

Безпосередньо з принципу згортання випливає існування такої множини. Нехай A — деяка множина, а $P(x)$ має вигляд $x \neq x$, тоді множина $\{a \in A \mid P(a)\} = \{a \in A \mid a \neq a\}$, очевидно, не має елементів. Із принципу об'ємності випливає, що може існувати лише одна множина, яка не має елементів. Ця множина називається **пустою множиною** і позначається $\varnothing$.

Підводячи перші підсумки, зазначимо, що множини задаються або за допомогою явного переліку своїх елементів (випадок скінченної множини з невеликою кількістю елементів), або за допомогою властивості, якій задовольняють її елементи (властивість також може виражати спосіб побудови елементів множини). Задання множини будемо називати ненадлишковим, якщо кожний її елемент входить у дану множину в єдиному екземплярі, і **надлишковим**, якщо хоча б один елемент цієї множини входить до її складу більш як в одному екземплярі (випадок мультимножини).

Більш детальне викладення основ теорії множин і її аксіоматику можна знайти в таких монографіях [16, 20, 21, 25, 36, 42].

1.2. Операції над множинами. Закони для операцій

Введемо символи $\Leftrightarrow$, $\exists x$, $\forall x$, $\Rightarrow$, які надалі будуть служити для скорочення виразів «*тоді і тільки тоді, коли*», «*існує x такий, що*», «*для кожного x*» і «*випливає*» відповідно.

Визначення 2. *Множина A називається* **підмножиною** *множини B ($A \subseteq B$), якщо всі елементи множини A є також елементами множини B ($A \subseteq B \Leftrightarrow (a \in A \Rightarrow a \in B)$). При цьому множина B називається* **надмножиною** *множини A.*

Тепер принцип об'ємності можна записати так:

$$A = B \Leftrightarrow (A \subseteq B \text{ і } B \subseteq A).$$

Визначення 3. *Введемо знак строгого включення $\subset$ для множин. $A \subset B$ означає для множин A і B, що $A \subseteq B$ і A не дорівнює B ($A \neq B$). Якщо $A \subset B$, то множина A називається* **власною підмножиною** *множини B, а множина B —* **власною надмножиною** *множини A.*

Приклад 1.1.3. Введемо до розгляду деякі спеціальні множини.

1. Розглянемо множину, яка буде часто фігурувати в подальшому. **Алфавітом** X називається скінченна непуста множина попарно різних між собою елементів, які називаються **літерами:** $X = \{x_1, x_2, x_3, \ldots, x_n\}$. **Словом** в алфавіті X називається довільна скінченна послідовність літер цього алфавіту. Множина всіх скінченних слів у алфавіті X позначається через $F(X)$ або X^*.

Довільна підмножина множини $F(X)$ називається **мовою в алфавіті** X. Наприклад,

а) нехай $X = \{a, b, c, d, \ldots, z\}$ складається з 26 літер англійського алфавіту. Довільна послідовність літер з X належить до $F(X)$. Отже, $F(X)$ включає слова *math*, *is*, *fun*, *men*, *amour* і т. д. Оскільки $F(X)$ включає множину слів *a*, *aa*, *aaa*, *aaaa*, *aaaaa*,..., то $F(X)$ є нескінченною множиною. Можна було б визначити американську мову L як підмножину $F(X)$, що складається зі слів сучасного видання *Webster's New Word Dictionary of the American Language*. Приклад цієї мови показує, що хоча L включає велику кількість елементів, але вона є скінченною множиною.

b) якщо $X = \{0, 1\}$, то множина B тих слів з множини $F(X)$, які починаються з 1, являє собою множину двійкових (бінарних) цілих додатних чисел, тобто $B = \{1, 10, 11, 100, 101, 110, 111, 1000, 1001, \ldots\}$.

Введемо до множини $F(X)$ деяке спеціальне слово, яке певною мірою аналогічне до пустої множини і яке називають *пустим словом*: пусте слово за означенням не включає жодного символу алфавіту X і позначається через e. Наприклад,

а) якщо $X = \{a\}$, то $F(X) = \{e, a, aa, aaa, aaaa, aaaaa, \ldots\}$;

b) якщо $X = \{a, b\}$, то $F(X) = \{e, a, b, aa, ab, ba, bb, aaa, aab, \ldots\}$;

c) якщо $X = \{0, 1, 2\}$, то $F(X) = \{e, 0, 1, 2, 00, 01, 02, 10, 11, 12, 20, 21, 22, 000, \ldots\}$.

Якщо $p \in F(X)$ є деяким слово, то **довжиною слова** p називається число літер алфавіту X у слові p, рахуючи кожне входження літери. Будемо позначати довжину слова p за допомогою $l(p)$. Наприклад, якщо $X = \{a, b\}$, то $l(aab) = l(baa) = 3$, а для пустого слова маємо $l(e) = 0$.

Конкатенацією двох слів p і q в алфавіті X називається слово pq, одержане в результаті приписування праворуч до слова p слова q. Очевидно, що для пустого слова справедливі такі рівності:

$$\forall p \in F(X)\ (ep = pe = p).$$

Нехай дано слово $q = uw$, де $u, w \in F(X)$. Слово u називається **початком** або **префіксом** слова q, а слово w — **кінцем** або **суфіксом** слова q. Слово p є **підсловом** слова q в алфавіті X, якщо

$$q = p'pq',$$

де p', q' деякі підходящі слова з $F(X)$. Якщо p' є словом найменшої можливої довжини, то входження слова p в слово q називається **першим.** Аналогічно можна означити друге входження слова, третє і т. д. Наприклад, якщо $X = \{a, b\}$, то

$$L = \{p \in F(X) : l(p) = 2\} = \{aa, ab, ba, bb\};$$

якщо $L' = \{p \in F(X) : l(p)$ є числом парним$\}$, то

$$L' = \{aa, ab, ba, bb, aaaa, aaab, aabb, abbb, ...\}.$$

Зазначимо, що L є підмножиною множини L'. Слово $p = baac$ є підсловом слова $q = abaacb$. Слова e, a, ab, aba, $abaac$, $abaacb$ є префіксами слова q, а слова e, b, cb, acb, $aacb$, $baacb$, $abaacb$ є суфіксами слова q.

2. Множина натуральних чисел $\{0, 1, 2,..., n,...\}$, яку позначають буквою N. Цю множину можна задати за допомогою принципу згортання з використанням поняття мови.

Нехай $A = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ — множина з десяти елементів, яка складає алфавіт. Словами в цьому алфавіті є довільна скінченна послідовність символів із A. Наприклад, $p = 0020034985700$ — слово в алфавіті A, а $p = 00a26543c$ не є словом в алфавіті A, оскільки до його запису входять символи (символи a, c), що не належать множині A. Тобто p буде словом в алфавіті A тоді і тільки тоді, коли кожний символ цього слова належить алфавіту A.

Тоді множина $N = \{p = xy...z | x, y,..., z \in A\}$ складає множину натуральних чисел. Зауважимо, що слово $p = 00...0$ і $p' = 0$ — одне й те саме число 0, так само, як і числа 00057 і 57. Отже, запис одного й того самого елемента множини N може бути різним. Для однозначності запису чисел необхідно поставити вимогу, щоб перший символ у

слові p, яке складається більше ніж з одного символа, був відмінний від нуля, тобто $N = \{p = xy\ldots z | x, y, \ldots, z \in A \text{ і } x \neq 0$ або 0, якщо $p = 0\}$.

3. Множина натуральних додатних чисел $N^+ = \{1, 2, \ldots, n, \ldots\}$ — це така множина слів у алфавіті A, серед яких немає слова вигляду $p = 0$.

4. Множина цілих чисел $Z = \{\ldots, -n, \ldots, -2, -1, 0, 1, 2, \ldots, n, \ldots\}$ — це множина слів у алфавіті $B = \{-, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ таких, що $Z = \{p = xy\ldots z | x, y, \ldots, z \in B$ і $x \neq 0$ і $y, \ldots, z \neq -$ або 0, якщо $p = 0\}$. Слово, першим символом якого є символ «–», називається від'ємним числом, а слово, першим символом якого є символ, відмінний від символа «–», називається додатним числом.

5. Множина раціональних чисел Q складається із множини всіх цілих чисел і всіх нескорочуваних раціональних дробів виду m/n, де $m, n \in Z, n \neq 0$, тобто $Q = \{m/n | m, n \in Z$ і $n \neq 0\}$.

6. Множина дійсних чисел D складається із множини всіх раціональних чисел і всіх ірраціональних чисел (чисел, які представляються у вигляді нескінченних неперіодичних десяткових дробів).

7. Для введених вище множин маємо такі включення $N^+ \subset N$, $N \subset Z, Z \subset Q, Q \subset D$ або $N^+ \subset N \subset Z \subset Q \subset D$.

8. Візьмемо довільну множину S. Очевидно, що коли $x \in S$, то $x \in S$ і тому $S \subseteq S$. Це означає, що довільна множина є своєю підмножиною. Для того і вводилися позначення $\subseteq$ і $\subset$.

9. Множина студентів юридичного факультету — власна підмножина множини всіх студентів університету.

10. Множина парних натуральних чисел є власною підмножиною множини всіх натуральних чисел. ♠

Покажемо, що пуста множина є підмножиною довільної множини A. Припустимо, що $\varnothing \subseteq A$ хибне. Це можливо лише тоді, коли існує хоча б один елемент із множини $\varnothing$, який не є елементом множини A. Але це неможливо, оскільки пуста множина не має елементів узагалі. Отже, $\varnothing \subseteq A$.

Введемо до розгляду ще одну важливу множину. Нехай U — деяка множина, тоді через $B(U)$ позначимо множину всіх підмножин множини U. Множину $B(U)$ називають **множиною-степенем** або **булеаном** множини U.

Приклад 1.1.4. Булеани множин $\varnothing$, $\{a\}$, $\{a, b\}$ $\{a, b, c\}$ ($a \neq b$, $a \neq c$, $b \neq c$) є відповідно такими множинами:

a) $B(\varnothing) = \{\varnothing\}$;

b) $B(\{a\}) = \{\varnothing, \{a\}\}$ (тобто булеан має два елементи);

c) $B(\{a, b\}) = \{\varnothing, \{a\}, \{b\}, \{a, b\}\}$ (тобто булеан має чотири елементи);

d) $B(\{a, b, c\}) = \{\varnothing, \{a\}, \{b\}, \{c\}, \{a, b\}, \{a, c\}, \{b, c\}, \{a, b, c\}\}$ (тобто булеан має вісім елементів).

Зазначимо, що коли множина S має n елементів, то її булеан $B(S)$ має 2^n елементів. Якщо множина S є нескінченною, то очевидно що множина $B(S)$ теж є нескінченною. ♠

Визначення 4. **Об'єднанням $A \cup B$ множин** *A і B називається множина, до складу якої входять ті і тільки ті елементи, які входять до складу хоча б однієї з цих множин, тобто* $A \cup B = \{a | a \in A$ *або* $a \in B\}$.

Приклад 1.1.5. 1. Нехай $A = \{1, 2, 3\}$, $B = \{1, 3, 4, 6\}$, тоді $A \cup B = \{1, 2, 3, 4, 6\}$.

2. Нехай P — множина всіх парних натуральних чисел, а H — множина всіх непарних натуральних чисел, тоді $P \cup H = N$, де N — множина натуральних чисел. ♠

Визначення 5. **Перетином $A \cap B$ множин** *A і B називається множина, елементами якої є елементи, що входять до складу як множини A, так і множини B, тобто* $A \cap B = \{a | a \in A$ *і* $a \in B\}$. *Якщо $A \cap B = \varnothing$, то множини A і B називаються такими, що* **не перетинаються.**

Приклад 1.1.6. 1. Нехай A, B, P, H означають множини з попереднього прикладу, тоді

а) $A \cap B = \{1, 3\}$; $P \cap H = \varnothing$; *b)* $N \cap H = H$; $N \cap P = P$.

2. Нехай A — множина прямих, які проходять через точку a деякої площини, а B — множина прямих, які проходять через точку c цієї самої площини. Тоді $A \cap B = \{l\}$, де l — пряма, яка проходить через точки a і c. ♠

Визначення 6. **Різницею $A \setminus B$ множин** *B і A називається множина* $\{a | a \in B$ *і* $a \notin A\}$. *Очевидно, що* $B \setminus A = B \setminus (A \cap B)$. *Якщо* $A \subseteq B$, *то $B \setminus A$ називається* **доповненням** *множини A в множині B і позначається A'_B або просто A', коли B можна визначити із контексту.*

Визначення 7. **Симетричною різницею $A \div B$ множин** *A і B називається множина* $(A \setminus B) \cup (B \setminus A)$, *тобто* $A \div B = (A \setminus B) \cup (B \setminus A)$.

З означення операції $A \div B$ очевидним чином випливає така рівність: $A \div B = (A \cup B) \setminus (A \cap B)$.

Приклад 1.1.7. 1. Множина $N \setminus P = H$, тобто $N \setminus P$ являє собою множину всіх непарних натуральних чисел. Навпаки, $N \setminus H = P$, а $P \setminus N = H \setminus N = \varnothing$.

2. Нехай $A = \{1, 2, 3\}$, $B = \{1, 3, 4, 5\}$, тоді $B \setminus A = \{1, 3, 4, 5\} \setminus \{1, 2, 3\} = \{4, 5\} = B \setminus (A \cap B) = \{1, 3, 4, 5\} \setminus \{1, 3\} = \{4, 5\}$, $A \div B = B \div A = \{2, 4, 5\}$.

Множина $N \div N^+ = \{0\}$, оскільки $N \cap N^+ = N^+$, а $N \div P = H$.

3. Нехай $A = \{n \in N : n \le 11\}$, $B = \{n \in N$: число n парне і $n \le 20\}$ і $E = \{n \in N$: число n парне$\}$, тоді маємо:

$A \cup B = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 14, 16, 18, 20\}$,

$A \cap B = \{0, 2, 4, 6, 8, 10\}$,

$A \setminus B = \{1, 3, 5, 7, 9, 11\}$,

$B \setminus A = \{12, 14, 16, 18, 20\}$,

$A \div B = \{1, 3, 5, 7, 9, 11, 12, 14, 16, 18, 20\}$.

Крім того:

$E \cap B = B$, $B \setminus E = \varnothing$,

$E \setminus B = \{n \in N$: число n парне і $n \ge 22\} = \{22, 24, 26, 28, 30, 32, \ldots\}$,

$A \div E = \{1, 3, 5, 7, 9, 11\} \cup \{n \in N$: число n парне і $n \ge 12\} = \{1, 3, 5, 7, 9, 11, 12, 14, 16, 18, 20, 22, \ldots\}$

4. Розглянемо інтервали [0,2] і (0,1] у множині дійсних чисел D. Тоді $(0, 1] \subseteq [0, 2]$, а отже, $(0,1] \cup [0, 2] = [0, 2]$, а також $(0,1] \cap [0, 2] = (0,1]$.

Крім того: $(0,1] \setminus [0, 2] = \varnothing$,

$[0, 2] \setminus (0,1] = (1, 2] \cup \{0\}$, а також $[0, 2] \setminus (0, 2) = \{0, 2\}$

5. Нехай $X = \{a, b\}$, $A = \{e, a, aa, aaa\}$, $B = \{e, b, bb, bbb\}$ і $C = \{p \in F(X) : l(p) \le 2\}$.

Тоді маємо:

$A \cup B = \{e, a, b, aa, bb, aaa, bbb\}$,

$A \cap B = \{e\}$,

$A \setminus B = \{a, aa, aaa\}$,

$B \setminus A = \{b, bb, bbb\}$,

$A \cap C = \{e, a, aa\}$

$B \setminus C = \{bbb\}$

$C \setminus A = \{b, ab, ba, bb\}$

$A \setminus X = \{e, aa, aaa\}$ ♠

Якщо розглядається деяка множина U і всі її підмножини, то множина U в цьому випадку називається **універсальною множиною** або **універсумом.**

Приклад 1.1.8. 1. Якщо універсумом є множина N, а множини A і E визначені так, як у прикладі 1.1.6 (3), то $A' = \{n \in N : n \ge 12\}$ і $E' = \{n \in N$: число n непарне$\}$.

2. Якщо універсальною множиною є D, то

$[0, 1]' = (-\infty, 0) \cup (1, \infty)$,

$(0, 1)' = (-\infty, 0] \cup [1, \infty)$,

$\{0, 1\}' = (-\infty, 0) \cup (0, 1) \cup (1, \infty)$.

Для довільного числа $a \in R$ маємо: $[a, \infty)' = (-\infty, a)$ і $(a, \infty)' = (-\infty, a]$. ♠

Введені операції над множинами називають **теоретико-множинними операціями** і ці операції задовольняють певним законам. Розглянемо основні з цих законів.

Теорема 1. *Для довільних підмножин A, B, C деякої універсальної множини U справедливі такі рівності*:

$M1)$ $A \cup B = B \cup A, A \cap B = B \cap A$ **(закони комутативності);**

$M2)$ $A \cup (B \cup C) = (A \cup B) \cup C$,

$A \cap (B \cap C) = (A \cap B) \cap C$ **(закони асоціативності);**

$M3)$ $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$,

$A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$ **(закони дистрибутивності);**

$M4)$ $A \cup A' = U$, $A \cap A' = \varnothing$;

$M5)$ $A \cup \varnothing = A$, $A \cap U = A$.

Доведення. Покажемо справедливість другого співвідношення із M2: $A \cap (B \cap C) = (A \cap B) \cap C$ шляхом включення в обидві сторони.

Нехай $a \in A \cap (B \cap C)$, тоді $a \in A$, $a \in B$, $a \in C \Rightarrow a \in (A \cap B)$ і $a \in C \Rightarrow a \in (A \cap B) \cap C$ і $A \cap (B \cap C) \subseteq (A \cap B) \cap C$.

Одержання оберненого включення виконується аналогічно, тільки у зворотному порядку.

Покажемо тим самим способом справедливість першого співвідношення із $M3$.

Нехай $a \in A \cup (B \cap C)$, тоді може мати місце один з таких випадків:

a) $a \in A$ і $a \notin B \cap C$;

b) $a \notin A$ і $a \in B \cap C$;

c) $a \in A$ і $a \in B \cap C$.

Розглянемо по черзі кожний з випадків.

У випадку a) маємо $a \in A \cup B$ і $a \in A \cup C$, оскільки $a \in A$, а це означає, що $a \in (A \cup B) \cap (A \cup C)$.

У випадку b) маємо $a \in A \cup B$ і $a \in A \cup C$, оскільки a є елементом як множини B, так і множини C, а це означає, що $a \in (A \cup B) \cap (A \cup C)$.

У випадку c) маємо $a \in A \cup B$ і $a \in A \cup C$, оскільки a є елементом як множини A, так і множин B і C, а це означає, що $a \in (A \cup B) \cap (A \cup C)$. Отже, у кожному з випадків маємо $A \cup (B \cap C) \subseteq (A \cup B) \cap (A \cup C)$.

Нехай тепер $a \in (A \cup B) \cap (A \cup C)$, тоді $a \in A \cup B$ і $a \in A \cup C$. Якщо $a \in A$, то $a \in A \cup (B \cap C)$. А якщо $a \notin A$, то $a \in B$ і $a \in C$ і тоді

$a \in B \cap C$. Отже, $(A \cup B) \cap (A \cup C) \subseteq A \cup (B \cap C)$. Разом з отриманим раніше включенням маємо потрібну рівність.

Решта співвідношень доводиться аналогічно і їх доведення пропонуються читачеві як вправи (див. вправу 12 у кінці параграфа). ■

Використовуючи закони асоціативності для операцій об'єднання і перетину множин, можна ввести такі скорочення. Вирази

$$A_1 \cup A_2 \cup \ldots \cup A_n = \bigcup_{i=1}^{n} A_i$$

означають об'єднання множин A_1, A_2,..., A_n, тобто сукупність тих елементів, які є елементами хоча б однієї із множин A_1, A_2,..., A_n, а вирази

$$A_1 \cap A_2 \cap \ldots \cap A_n = \bigcap_{i=1}^{n} A_i$$

означають перетин множин, тобто сокупність тих предметів, які є елементами кожної із множин A_1, A_2,..., A_n. Ці скорочення розширюють і на випадок нескінченної сукупності множин, тобто

$$A_1 \cup A_2 \cup \ldots \cup A_k \cup \ldots = \bigcup_{i=1}^{\infty} A_i ,$$

$$A_1 \cap A_2 \cap \ldots \cap A_k \cup \ldots = \bigcap_{i=1}^{\infty} A_i .$$

Визначення 8. *Якщо множина A є об'єднанням своїх підмножин A_1, A_2,..., A_n,..., то сукупність підмножин $\{A_1, A_2, ..., A_n, ...\}$ називається* **покриттям** *множини A.*

Якщо покриття множини A є таким, що $A_i \cap A_j = \varnothing$ при $i \neq j$, то таке покриття називається **розбиттям** *множини A, а підмножини A_j —* **класами** *цього розбиття, $i = 1, 2, \ldots, n, \ldots$.*

Приклади покриття і розбиття множини

Нехай A — множина всіх студентів деякого вузу X, які його закінчили, а A_i — підмножина тих студентів вузу X, які закінчили i-й факультет цього вузу.

Зрозуміло, що сукупність підмножин A_1, A_2,..., A_k є, очевидно, покриттям множини A, оскільки існує можливість, що деяка людина з множини студентів A закінчила два факультети даного вузу, і така людина потрапляє в дві підмножини сукупності $\{A_i\}$, $i = 1, 2, ..., k$.

Якщо ж узяти сукупність усіх студентів вузу X, які навчаються в даний час, то сукупність студентів A_1, A_2,..., A_k є, очевидно, покриттям множини всіх студентів даного вузу, які навчаються в даний час ♠

Окрім тотожностей, які були наведені в теоремі 1, існують і інші важливі тотожності для операцій над множинами.

Теорема 2. *Для довільних підмножин A і B деякої універсальної множини U мають місце такі рівності*:

$M6)$ $A \cup A = A, A \cap A = A$ **(закон ідемпотентності);**

$M7)$ $A \cup (A \cap B) = A, A \cap (A \cup B) = A$ **(закон поглинання);**

$M8)$ $A \cup U = U, A \cap \varnothing = \varnothing$;

$M9)$ $(\varnothing)' = U, U' = \varnothing, (A')' = A$;

$M10)$ $(A \cup B)' = A' \cap B', (A \cap B)' = A' \cup B'$ **(закони де Моргана)**

Доведення. Покажемо, наприклад, справедливість тотожностей $M6$) і $M8$). Виходячи із законів $M1$)—$M5$), можемо записати

$$A = A \cup \varnothing = A \cup (A \cap A') = (A \cup A) \cap (A \cup A') = (A \cup A) \cap U = A \cup A,$$
$$A = A \cap U = A \cap (A \cup A') = (A \cap A) \cup (A \cap A') = (A \cap A) \cup \varnothing = A \cap A.$$

Далі, користуючись тільки що встановленими тотожностями, маємо:

$$A \cup U = A \cup (A \cup A') = (A \cup A) \cup A' = A \cup A' = U,$$
$$A \cap \varnothing = A \cap \{A \cap A') = (A \cap A) \cap A' = A \cap A' = \varnothing.$$

Доведення решти тотожностей пропонуються читачеві як вправи.■

Користуючись тотожностями, наведеними в теоремах 1, 2, можна виконувати спрощення складних виразів, які містять множини, аналогічно тому, як проводяться спрощення виразів в елементарній алгебрі. Розглянемо приклади.

Приклад 1.1.9.

1. $(A \cap B \cap C) \cup (A' \cap B \cap C) \cup B' \cup C = [(A \cup A') \cap B \cap C] \cup B' \cup C' =$
$(U \cap B \cap C) \cup B' \cup C' = (B \cap C) \cup B' \cup C' = (B \cap C) \cup (B \cap C)' = U.$

2. $(A \cap B \cap C \cap D') \cup (A' \cap C) \cup (B' \cap C) \cup (C \cap D) = (A \cap B \cap C \cap D') \cup$
$\cup [(A' \cup B' \cup D) \cap C] = [(A \cap B \cap D') \cup (A \cap B \cap D')'] \cap C = U \cap C = C.$

3. $(A \cap B')' \cup B = (A' \cup B'') \cup B = A' \cup (B \cup B) = A' \cup B.$

4. $(A \cap B) \cup (A' \cap C) \cup (B \cap C) = (A \cap B) \cup (A' \cap C) \cup (B \cap (A \cup A') \cap C =$
$= (A \cap B) \cup (A' \cap C) \cup (A \cap B \cap C) \cup (A' \cap B \cap C) = (A \cap B) \cap (U \cup C) \cup$
$\cup (A' \cap C) \cap (U \cup B) = ((A \cup B) \cap U) \cup ((A' \cap C) \cap U) = (A \cap B) \cup (A' \cap C).$

5. $(A \cup B) \cap (A \cup C) = A \cup (B \cap C)$ на підставі закону дистрибутивності, а звідси випливає справедливість такої рівності:

$$(A \cup B) \cap (A \cup B') = A \cup (B \cap B') = A \cup \varnothing = A \spadesuit$$

1.3. Декартовий добуток множин. Відношення

Нехай A і B — дві множини. Розглянемо множину упорядкованих пар вигляду

$$C = \{(a, b) | a \in A, b \in B\}.$$

Ця множина називається **декартовим добутком множин** A і B і позначається $A \times B$. Якщо множини A і B скінченні і складаються відповідно із m і n елементів, то очевидно, що C складається із $m \cdot n$ елементів (див. теорему 17).

Самостійний інтерес являє собою випадок, коли множини A і B рівні між собою: $A = B$. Для його розгляду введемо поняття упорядкованої пари.

Упорядкованою парою елементів множини A будемо називати об'єкт (a, a'), що складається з двох, не обов'язково різних, елементів a і a' множини A, для яких вказано, котрий із них потрібно вважати першим, а котрий — другим. Так, наприклад, якщо $A = \{1, 2, 3, 4, 5\}$, то упорядковані пари (3, 4) і (4, 3) слід вважати різними, оскільки в першій парі першим елементом є 3 і другим елементом — 4, а в другій парі — навпаки. Упорядкованими парами також є пари (1, 1), (2, 2), (3, 3), (4, 4), (5, 5).

Множина $C = \{(a, a') | a, a' \in A\}$ усіх упорядкованих пар (a, a') елементів із множини A називається **декартовим квадратом** множини A і позначається A^2 [21, 9, 3].

Поняття упорядкованої пари можна розширити на упорядковані трійки елементів (a_1, a_2, a_3) із A, упорядковані четвірки (a_1, a_2, a_3, a_4) із A і т. д. Множину всіх упорядкованих трійок елементів із множини A будемо позначати A^3, множину всіх упорядкованих четвірок елементів із множини A — A^4 і т. д. Загалом, упорядкована n-ка елементів із множини A — це n не обов'язково різних між собою елементів із A, заданих у певній послідовності $(a_1, a_2, ..., a_n)$ так, що

$$(a_1, a_2, ..., a_n) = (a'_1, a'_2, ..., a'_n)$$

тоді і тільки тоді, коли $a_1 = a'_1$, $a_2 = a'_2$, ..., $a_n = a'_n$.

Для того щоб відрізняти упорядковані пари, трійки і т. д. від неупорядкованих, введемо таке позначення: якщо A — деяка множина, то $A^{(2)}$, $A^{(3)}$, ... будуть відповідно означати множину неупорядкованих пар елементів, трійок елементів і т. д. із множини A, тобто $A^{(2)}$ являє собою множину всіх двоелементних підмножин множини A, $A^{(3)}$ — множину всіх триелементних підмножин множини A і т. д.

Наведені вище означення декартового добутку двох множин і декартового квадрату множини можна звичайним способом узагальнити і на випадок довільної скінченної сукупності множин.

Визначення 9. **Декартовим добутком** $A_1 \times A_2 \times \ldots \times A_n$ *множин* $A_1, A_2, \ldots, A_n$ *називається сукупність послідовностей (тобто сукупність упорядкованих n-ок елементів) вигляду* $(a_1, a_2, \ldots, a_n)$, *де* $a_i \in A_i$, $1 \le i \le n$.

Елементи декартового добутку часто називають *кортежами*.

Визначення 10. *Довільна підмножина R множини* $A_1 \times A_2 \times \ldots \times A_n$ *називається* **відношенням,** *заданим або визначеним на множинах* $A_1, A_2, \ldots, A_n$. *Якщо* $A_1 = A_2 = \ldots = A_n = A$, *то декартовий добуток* $A_1 \times A_2 \times \ldots \times A_n$ *називається* **декартовим добутком *n*-ого степеня множини** $A(A^n)$, *а відношення R, задане на множинах* $A_1, A_2, \ldots, A_n$, — ***n*-арним відношенням** *на множині A.*

Зокрема, при $n = 1$ відношення називається *унарним*, при $n = 2$ — *бінарним*, при $n = 3$ — *тернарним*.

Якщо $(a_1, a_2, \ldots, a_n) \in R$, то говорять, що елементи a_i $(i = 1, 2, \ldots, n)$ знаходяться у відношенні R між собою або що відношення R **істинне** для $a_1, a_2, \ldots, a_n$. Якщо $(a_1, a_2, \ldots, a_n) \notin R$, то говорять, що R **хибне** для $a_1, a_2, \ldots, a_n$.

Оскільки відношення, задані на $A_1, A_2, \ldots, A_n$, суть підмножини множини $A_1 \times A_2 \times \ldots \times A_n$, то для них звичайним чином вводяться операції об'єднання, перетину, різниці і доповнення:

$(a_1, a_2, \ldots, a_n) \in R \cup R_1 \Leftrightarrow (a_1, a_2, \ldots, a_n) \in R$ або $(a_1, a_2, \ldots, a_n) \in R_1$;
$(a_1, a_2, \ldots, a_n) \in R \cap R_1 \Leftrightarrow (a_1, a_2, \ldots, a_n) \in R$ і $(a_1, a_2, \ldots, a_n) \in R_1$;
$(a_1, a_2, \ldots, a_n) \in R \setminus R_1 \Leftrightarrow (a_1, a_2, \ldots, a_n) \in R$ і $(a_1, a_2, \ldots, a_n) \notin R_1$;
$(a_1, a_2, \ldots, a_n) \in R'$ в $A_1 \times A_2 \times \ldots \times A_n \Leftrightarrow (a_1, a_2, \ldots, a_n) \notin R$.

Часто R' позначають через $\neg R$ і називають **запереченням** відношення R.

Найчастіше як у теорії, так і на практиці використовуються бінарні відношення. Якщо R — бінарне відношення, то запис aRb означає, що $(a, b) \in R$, тобто що R істинне для a, b.

Визначення 11. *Нехай дано бінарні відношення* $R \subseteq A \times B$, $R_1 \subseteq B \times C$. *Відношення* $R^{-1} = \{(b, a) \mid aRb\}$, *задане на множині* $B \times A$, *називають* **оберненим** *до відношення R, а відношення* $R * R_1 = \{(a, c) \mid a \in A, c \in C$ і $(\exists b \in B)\ aRb$ і $bR_1c\}$, *задане на множині* $A \times C$, — **добутком** *або* **суперпозицією** *відношень R і* R_1.

Приклад 1.1.10. Нехай $A = \{7, 3, 2, 4\}$, $B = \{3, 2, 9, 1\}$, $C = \{2, 0, 1, 7\}$ і $R \subseteq A \times B$, $R_1 \subseteq B \times C$ є відношеннями, визначеними таким чином:

$$mRn \Leftrightarrow m - n \text{ є числом парним},$$
$$kR_1l \Leftrightarrow k - l = 2.$$

Побудувати відношення R^{-1}, R_1^{-1}, $R * R_1$ і $R_1 * R$.

Відповідно до означення відношень R і R_1, одержуємо:

$$R = \{(7, 3), (7, 9), (7, 1), (3, 3), (3, 9), (3, 1), (2, 2), (4, 2)\}$$
$$R_1 = \{(3, 1), (2, 0), (9, 7)\}.$$

З означення відношення R^{-1} випливає, що пара $(x, y) \in R^{-1}$, коли пара (y, x) належить до відношення R. Оскільки пара $(7, 3) \in R$, то пара $(3, 7)$ є елементом R^{-1} і тоді

$$R^{-1} = \{(3, 7), (9, 7), (1, 7), (3, 3), (9, 3), (1, 3), (2, 2), (2, 4)\},$$
$$R_1^{-1} = \{(1, 3), (0, 2), (7, 9)\}.$$

Згідно з означенням відношення $R * R_1$, потрібно взяти пару (x, y) з відношення R і знайти пару (y, z), яка є елементом відношення R_1. Потім створити пару (x, z) і включити її до елементів відношення $R * R_1$.

Візьмемо пару $(7, 3)$ з R. У множині R_1 їй відповідає тільки пара $(3, 1)$. Будуємо пару $(7, 1)$ як елемент відношення $R * R_1$. Аналогічно будуємо пари для елементів $(7, 9)$, $(7, 1)$ і т. д. У результаті одержуємо такі відношення:

$$R * R_1 = \{(7, 1), (7, 7), (3, 1), (3, 7), (2, 0), (4, 0)\}.$$
$$R_1 * R = \{(9, 3), (9, 9), (9,1)\}. \spadesuit$$

Розглянемо властивості бінарних відношень.

Теорема 3. *Якщо R, R_1, R_2 — бінарні відношення, задані на множині A, то*

а) $(R_1 \cup R_2) * R = (R_1 * R) \cup (R_2 * R)$; $R_1 \subseteq R_2 \Rightarrow R_1 * R \subseteq R_2 * R$;

б) $(R^{-1})^{-1} = R$; $R \subseteq R_1 \Rightarrow R^{-1} \subseteq R_1^{-1}$;

в) $(R * R_1)^{-1} = R_1^{-1} * R^{-1}$;

г) $(R \cap R_1)^{-1} = R^{-1} \cap R_1^{-1}$;

д) $(R * R_1) * R_2 = R * (R_1 * R_2)$.

Доведення. а) Якщо $(a, b) \in (R_1 \cup R_2) * R$, то існує елемент $c \in A$ такий, що $(a, c) \in R_1 \cup R_2$ і $(c, b) \in R$. Отже, $(a, c) \in R_1$ або $(a, c) \in R_2$

і $(c, b) \in R$. Звідси маємо, що $(a, b) \in R_1 * R$ або $(a, b) \in R_2 * R$, тобто $(a, b) \in (R_1 * R) \cup (R_2 * R)$. Отже, $(R_1 \cup R_2) * R \subseteq (R_1 * R) \cup (R_2 * R)$.

Обернене включення доводиться аналогічно.

Друга частина твердження випливає з того, що коли $R_1 \subseteq R_2$, то $R_1 \cup R_2 = R_2$, звідки маємо (зважаючи на вищедоведене) $(R_1 \cup R_2) * R = R_1 * R \cup R_2 * R = R_2 * R$, тобто $R_1 * R \subseteq R_2 * R$.

б) $(a, b) \in R^{-1} \Leftrightarrow (b, a) \in (R^{-1})^{-1} \Leftrightarrow (b, a) \in R$. Звідки випливає, що $R = ((R^{-1})^{-1})$.

Для доведення другої частини зауважимо, що $(a, b) \in R \Leftrightarrow (b, a) \in R^{-1} \Rightarrow (a, b) \in R \Rightarrow (a, b) \in R_1 \Rightarrow (b, a) \in R^{-1} \Rightarrow (b, a) \in R_1^{-1}$, тобто $R^{-1} \subseteq R_1^{-1}$.

в) $(a, b) \in (R * R_1)^{-1} \Leftrightarrow (b, a) \in R * R_1 \Rightarrow \exists c \in A\ (b, c) \in R$ і $(c, a) \in R_1$. Але тоді $(c, b) \in R^{-1}$ і $(a, c) \in R_1^{-1} \Rightarrow (a, b) \in R_1^{-1} * R^{-1}$, тобто $(R * R_1)^{-1} \subseteq \subseteq (R_1^{-1}) * (R^{-1})$.

Обернене включення доводиться аналогічно.

г) $(a, b) \in (R \cap R_1)^{-1} \Leftrightarrow (b, a) \in R \cap R_1 \Leftrightarrow (b, a) \in R$ і $(b, a) \in R_1 \Leftrightarrow \Leftrightarrow (a, b) \in R^{-1}$ і $(a, b) \in R_1^{-1}$, тобто $(R \cap R_1)^{-1} = R^{-1} \cap R_1^{-1}$.

д) Нехай $(a, d) \in (R * R_1) * R_2$, тоді існує $c \in A$ такий, що $(a, c) \in R * R_1$ і $(c, d) \in R_2$. Отже, існує b такий, що $(a, b) \in R$, $(b, c) \in R_1$ і $(c, d) \in R_2$, а це означає, що $(b, d) \in R_1 * R_2$ і $(a, d) \in R * (R_1 * R_2)$, тобто $(R * R_1) * R_2 \subseteq R * (R_1 * R_2)$. Обернене включення доводиться аналогічно. ■

Перейдемо до розгляду прикладів найбільш важливих бінарних відношень.

1.4. Бінарні відношення та їх основні властивості

Відношення тотожності. Бінарне відношення **тотожності**, задане на множині A, складається в точності з усіх пар вигляду (a, a), де $a \in A$, і позначається через i_A або просто i, якщо A фіксовано. Пари вигляду (a, a) називають *діагональними*, а відношення i_A — **діагоналлю**. Очевидно, що для довільного бінарного відношення R, визначеного на множині A, має місце рівність $i_A * R = R * i_A = R$.

Рефлексивні відношення. Бінарне відношення R, задане на множині A, називається **рефлексивним**, якщо $i_A \subseteq R$, тобто коли воно включає діагональ.

Прикладом рефлексивних відношень можуть служити такі бінарні відношення:

— пряма x паралельна прямій y у площині π;

— студент x — ровесник студента y.

Дійсно, у першому випадку з елементарної геометрії відомо, що дві прямі, які лежать в одній площині, паралельні, якщо вони або збігаються, або не мають ні однієї спільної точки, скільки б їх не продовжували. Оскільки пряма x збігається сама із собою, то пара (x, x) належить даному відношенню.

У другому випадку очевидно, що кожний студент — сам собі ровесник.

Іррефлексивні відношення. Бінарне відношення R називається **іррефлексивним,** якщо aRa хибне для довільного елемента $a \in A$. Наприклад, відношення $a < a$ на множині дійсних чи раціональних чисел хибне для кожного числа a.

Симетричні відношення. Бінарне відношення R, задане на множині A, називається **симетричним,** якщо із aRb випливає bRa $(R \subseteq R^{-1})$.

Прикладом симетричних відношень можуть служити такі бінарні відношення:

— пряма x перпендикулярна прямій y у площині π;

— студент x є сусідом по парті студента y.

Дійсно, у першому випадку з елементарної геометрії відомо: якщо пряма x перпендикулярна до прямої y, то і пряма y перпендикулярна до прямої x.

У другому ж випадку кожний студент може впевнитися в тому, що коли студент y є його сусідом по парті, то у студента y є сусідом він сам. Зауважимо, що наведені відношення не є рефлексивними.

Транзитивні відношення. Бінарне відношення R, задане на множині A, називається **транзитивним,** якщо із aRb і bRc випливає aRc $(R^2 \subseteq R)$.

Прикладом транзитивних відношень можуть служити такі бінарні відношення:

— місто x пов'язане з містом y шосейною дорогою;

— студент x є ровесником студента y;

— трикутник x подібний до трикутника y;

— дійсне число x більше ніж дійсне число y.

Дійсно, у першому випадку, якщо між містами x і y є шосейна дорога і між містами y і z також є шосейна дорога, то зрозуміло, що між містами x і z теж є шосейна дорога (яка, наприклад, пролягає через місто y).

В останніх трьох випадках транзитивність очевидна.

Антисиметричні відношення. Бінарне відношення R, задане на множині A, називається **антисиметричним,** якщо із aRb і bRa випливає $a = b$ $(R \cap R^{-1} \subseteq i_A)$.

Прикладом антисиметричного відношення може служити бінарне відношення включення для множин, тобто відношення «множина A є підмножиною множини B».

Справді, якщо $A \subseteq B$ і $B \subseteq A$, то звідси випливає, що множини A і B рівні між собою з огляду на аксіому об'ємності.

1.4.1. Відношення еквівалентності

Бінарне відношення R, задане на множині A, називається **відношенням еквівалентності** або просто **еквівалентністю** на A, якщо воно рефлексивне, симетричне і транзитивне, тобто якщо для довільних елементів a, b, c із A мають місце такі властивості:

а) aRa ($i_A \subseteq R$) (рефлексивність);

б) $aRb \Rightarrow bRa$ ($R \subseteq R^{-1}$) (симетричність);

в) aRb і $bRc \Rightarrow aRc$ ($R^2 \subseteq R$) (транзитивність),

де i_A — відношення тотожності, а $R^2 = R * R$.

Неважко показати, що умови а), б), в) еквівалентні таким $i_A \subseteq R$, $R = R^{-1}$, $R^2 = R$ (див. вправу 32 в кінці параграфа).

Приклад 1.1.11.

1. Для чисел $m, n \in Z$ визначимо: mRn тоді і тільки тоді, коли $m - n$ є числом непарним. Відношення R є симетричним, але воно нерефлексивне і нетранзитивне. Отже, $\neg(mRm)$ для всіх $m \in Z$. Аналогічно із mRn і nRp завжди випливає $\neg(mRp)$.

2. Для чисел $m, n \in N^+$ визначимо: $mR_2n \Leftrightarrow rest\,(m, 2) = rest\,(n, 2)$, де $rest\,(x, 2)$ означає остачу від ділення x на 2. Загалом для кожного $n \in N^+$ існує в точності одна пара цілих чисел p і r, яка задовольняє умову: $n = p \cdot q + r$ і $0 \le r < p$. Числа p і r називаються відповідно **часткою** і **остачею** від ділення числа n на p. Якщо запишемо $n/p = q + r/p$, то число q називається **цілою частиною** числа n/p і позначається символом $[n/p]$, а число r/p є **дробовою частиною** числа n/p. Покажемо, що відношення R є відношенням еквівалентності:

а) $mRm \Leftrightarrow rest\,(m, 2) = rest\,(m, 2)$ очевидно має місце;

b) $mRn \Leftrightarrow rest\,(m, 2) = rest\,(n, 2) \Leftrightarrow rest\,(n, 2) = rest\,(m, 2) \Leftrightarrow nRm$ теж правильне;

c) mRn і $nRk \Leftrightarrow rest\,(m, 2) = rest\,(n, 2) = rest\,(k, 2) \Rightarrow rest\,(m, 2) = rest\,(k, 2) \Rightarrow mRk$.

Звідси випливає, що відношення R є відношенням еквівалентності. ♠

Відношення еквівалентності, задане на множині A, тісно пов'язане з розбиттям множини A на класи. Цей зв'язок виражається такими твердженнями.

Лема 1. *Довільне розбиття множини A на класи визначає на множині A відношення еквівалентності.*

Доведення. Нехай $a, b \in A$, покладемо $aRb \Leftrightarrow a$ і b лежать в одному класі розбиття. Покажемо, що одержане бінарне відношення є відношенням еквівалентності.

aRa, оскільки a лежить у деякому класі розбиття.

$aRb \Rightarrow a, b \in K$ — деякий клас розбиття, але тоді і $b, a \in K \Rightarrow bRa$.

$aRb, bRc \Rightarrow a, b, c \in K \Rightarrow a, c \in K \Rightarrow aRc$. ■

Лема 2. *Довільне відношення еквівалентності R, визначене на множині A, задає розбиття множини A на класи.*

Доведення. Назвемо класом елемента a множину $K(a) = \{x \in A | aRx\}$. Із рефлексивності відношення R випливає, що $a \in K(a)$, тобто система класів $K(a)$ $(a \in A)$ покриває всю множину A. Далі симетричність відношення R показує, що коли $b \in K(a)$, то $a \in K(b)$, а транзитивність відношення R приводить до того, що якщо $b \in K(a)$, то із того, що $c \in K(b)$ випливає що $c \in K(a)$, тобто $K(b) \subseteq K(a)$. Але якщо $a \in K(b)$, то $K(a) \subseteq K(b)$, отже, $K(a) = K(b)$. Звідси випливає, що кожний клас визначається довільним своїм елементом. Якщо $K(a) \cap \cap K(b) \neq \varnothing$, то існує такий елемент c із цього перетину, що cRa і cRb і тоді $K(a) = K(b) = K(c)$, тобто класи $K(a)$ і $K(b)$ збігаються. Таким чином, доведено, що система всіх різних класів виду $K(a)$ є розбиттям множини A. ■

Очевидно, що перехід від розбиття S множини A, яке визначається відношенням R, до відношення еквівалентності R, а після того перехід від відношення R до розбиття множини A, знову приводить до розбиття S. Отже, має місце така теорема.

Теорема 4. *Між розбиттями множини на класи і відношеннями еквівалентності, заданими на цій множині, існує взаємно однозначна відповідність* [21].

Якщо R — еквівалентність на A, то класи розбиття, визначені відношенням R, називають **класами еквівалентності** відношення R, а множину всіх класів розбиття — **фактор множиною** множини A і позначають її A/R. Число класів еквівалентності відношення еквівалентності R називається **індексом** множини A. Якщо число класів еквівалентності скінченне, то множина A називається **множиною скінченного індексу.**

Приклад 1.1.12. Нехай Z є множиною цілих чисел і mRn тоді і тільки тоді, коли число $m - n$ парне. Чи є відношення R відношенням

еквівалентності? Якщо так, то скільки класів еквівалентності воно має?

Покажемо, що відношення R є відношенням еквівалентності.

Рефлексивність. mRm означає, що $m - m = 0$ є числом парним, а це правильно, оскільки 0 є числом парним.

Симетричність. mRn означає, що $m - n$ є числом парним, але звідси випливає, що і число $n - m = -(m - n)$ теж є парним, а це означає, що nRm.

Транзитивність. mRn і nRk означає, що числа $m - n$ і $n - k$ є парними. Число $m - n$ є парним якщо

a) m і n є числами парними або

b) m і n є числами непарними.

Якщо маємо випадок a), то число k теж має бути парним, оскільки різниця $n - k$ є числом парним і число n парне. Тоді різниця $m - n$ також є числом парним.

Якщо маємо випадок b), то число k теж має бути непарним, оскільки різниця $n - k$ є числом парним, а n є числом непарним. Тоді різниця $m - k$ є числом парним, оскільки m і k є числами непарними.

Звідси отримуємо, що відношення R є відношенням еквівалентності. Відношення R у множині цілих чисел Z має два класи еквівалентності. До одного класу належать усі парні числа, а до другого — усі непарні. Звідси випливає, що фактор-множина Z/R відносно даного відношення еквівалентності є множиною індексу два, тобто фактор-множиною скінченного індексу. ♠

З'ясуємо, які операції, що застосовуються до відношень еквівалентності заданих на деякій множині A, дають у результаті знову відношення еквівалентності, а які ні.

Теорема 5. *Якщо R, R_1 — відношення еквівалентності, що задані на множині A, то*

a) R^{-1} *— відношення еквівалентності на A;*

b) $R * R_1$ *— відношення еквівалентності на $A \Leftrightarrow R * R_1 = R_1 * R$, тобто коли відношення R і R_1 можна міняти місцями;*

c) $R \cap R_1$ *— відношення еквівалентності на A;*

d) R' *не є відношенням еквівалентності на A.*

Доведення. a) Оскільки R — відношення еквівалентності, то $R^{-1} = R$ і, отже, R^{-1} — теж відношення еквівалентності.

b) Якщо $R * R_1$ — відношення еквівалентності, то за доведеним вище $(R * R_1)^{-1}$ — теж відношення еквівалентності і зважаючи на теорему 3, $R * R_1 = (R * R_1)^{-1} = (R_1^{-1}) * (R^{-1}) = R_1 * R$.

Навпаки, якщо $R * R_1 = R_1 * R$, то із того, що xRx і xR_1x випливає $xR * R_1x$, і, отже, $R * R_1$ рефлексивне. Далі із того, що $(R * R_1)^{-1} = R_1^{-1} * R^{-1} = R_1 * R = R * R_1$ випливає симетричність $R * R_1$. І, нарешті, $(R * R_1) * (R * R_1) = R * R * R_1 * R_1 = R * R_1$, зважаючи на те, що їх можна міняти місцями, транзитивності і асоціативності множення відношень (див. теорему 3). Отже, $R * R_1$ — транзитивне.

c) Оскільки R і R_1 — еквівалентності на A, то $i_A \subseteq R$ і $i_A \subseteq R_1$, звідки випливає, що $i_A \subseteq R \cap R_1$.

Зважаючи на теорему 3 і те, що R і R_1 — еквівалентності, маємо $(R \cap R_1)^{-1} = (R^{-1}) \cap (R_1^{-1}) = R \cap R_1$. Отже, відношення $R \cap R_1$ — рефлексивне і симетричне. Покажемо транзитивність.

Нехай $(a, b) \in R \cap R_1$ і $(b, c) \in R \cap R_1$, тоді $(a, b) \in R$, $(a, b) \in R_1$ і $(b, c) \in R$, $(b, c) \in R_1$. Із $(a, b) \in R$, $(b, c) \in R$ випливає, що $(a, c) \in R$, а із $(a, b) \in R_1$, $(b, c) \in R_1$ випливає, що $(a, c) \in R_1$. Звідки одержуємо, що $(a, c) \in R \cap R_1$, тобто $R \cap R_1$ — транзитивне.

d) Із того, що R — еквівалентність, випливає, що $i_A \subseteq R$, але тоді i_A не може бути підмножиною множини R', тобто R' не є відношенням еквівалентності. ■

Об'єднання відношень еквівалентності загалом не завжди буде відношенням еквівалентності, як показує наступна теорема.

Теорема 6. *Об'єднання $R \cup R_1$ відношень еквівалентності R і R_1 є еквівалентністю тоді і тільки тоді, коли перетин довільного класу еквівалентності по R з довільним класом еквівалентності по R_1 або збігається з одним із них або пустий. Якщо $R \cup R_1$ — еквівалентність, то $R \cup R_1 = R * R_1$* [25].

1.4.2. Замикання відношень

Рефлексивне замикання

Визначення 12. *Відношення R_i називається* ***рефлексивним замиканням*** *бінарного відношення R на множині A, якщо $R_i = R \cup i_A$, де i_A — відношення тотожності (діагональ) на множині A.*

Безпосередньо з цього означення випливає таке твердження.

Теорема 7. *Нехай R деяке бінарне відношення на множині A, тоді*

a) $(R_i)_i = R_i$;

b) $R_i = R$ *тоді і тільки тоді, коли R рефлексивне.*

Доведення. а) Відповідно до означення рефлексивного замикання та законів асоціативності й ідемпотентності для об'єднання множин, одержуємо

$$(R_i)_i = R_i \cup i_A = (R \cup i_A) \cup i_A = R \cup (i_A \cup i_A) = R \cup i_A = R_i.$$

b) Якщо R рефлексивне відношення, то $i_A \subseteq R$ і тоді $i_A \cup R = R$ (див. вправу 1 в кінці розділу). Якщо $R_i = R$, то $i_A \cup R = R$, а це означає, що $i_A \subseteq R$. Звідси випливає, що відношення R є рефлексивним. ■

Симетричне замикання

Визначення 13. *Відношення R_s називається* ***симетричним замиканням*** *бінарного відношення R на множині A, якщо $R_s = R \cup R^{-1}$, тобто якщо $(a, b) \in R$, то $(a, b) \in R_s$ і $(b, a) \in R_s$.*

Безпосередньо з цього означення випливає таке твердження.

Теорема 8. *Нехай R — деяке бінарне відношення на множині A, тоді*

a) $(R_s)_s = R_s$;

b) $R_s = R$ *тоді і тільки тоді, коли відношення R симетричне;*

c) *якщо відношення R рефлексивне, то рефлексивне й відношення R_s.*

Доведення. а) Відповідно до означення симетричного замикання і законів асоціативності, комутативності та ідемпотентності для об'єднання множин, одержуємо

$$(R_s)_s = R_s \cup R_s^{-1} = (R \cup R^{-1}) \cup (R^{-1} \cup (R^{-1})^{-1}) = (R \cup R^{-1}) \cup (R^{-1} \cup R) =$$
$$= (R \cup R^{-1}) \cup (R \cup R^{-1}) = R_s \cup R_s = R_s.$$

b) Якщо відношення R симетричне, то $R \subseteq R^{-1}$ і тоді, згідно з теоремою 3, маємо $R^{-1} \subseteq (R^{-1})^{-1} = R$. А це означає, що $R = R^{-1}$. Використовуючи цю рівність і закон ідемпотентності для об'єднання множин, одержуємо $R_s = R \cup R^{-1} = R \cup R = R$. Якщо $R_s = R$, то відношення R симетричне, оскільки симетричне відношення R_s.

c) Якщо відношення R рефлексивне, то $i_A \subseteq R$ і $i_A = i_A^{-1} \subseteq R^{-1}$, а тоді $i_A \subseteq R \cup R^{-1} = R_s$. А це означає, що відношення R_s рефлексивне. ■

Транзитивне замикання

Визначення 14. *Відношення R_t називається* ***транзитивним замиканням*** *бінарного відношення R на множині A, якщо $R_t = R \cup R^2 \cup R^3 \cup \ldots \cup R^n \cup \ldots$, тобто $(a, b) \in R_t$ тоді і тільки тоді, коли існують елементи $a_1 = a, a_2, \ldots, a_n = b \in A$ такі, що $(a_1 R a_2, a_2 R a_3, \ldots, a_{n-1} R a_n)$.*

Використовуючи означення і певні властивості транзитивного відношення, можемо довести таке твердження.

Теорема 9. *Нехай R — деяке відношення на множині A, тоді*

a) $R_t = R$ *тоді і тільки тоді, коли відношення R транзитивне*;

b) $(R_t)_t = R_t$;

c) *якщо відношення R рефлексивне, то відношення R_t теж рефлексивне*;

d) *якщо відношення R симетричне, то відношення R_t теж симетричне*;

e) *якщо відношення R транзитивне, то відношення R_i теж транзитивне*;

f) *якщо множина A має n елементів, то* $R_t = R \cup R^2 \cup \ldots \cup R^n = \bigcup_{k=1}^{n} R^k$.

Доведення. a) З означення транзитивності відношення R випливає, що $R^2 \subseteq R$. З теореми 3 маємо $R^2 \subseteq R \Rightarrow R^3 \subseteq R^2 \Rightarrow R^4 \subseteq R^3 \ldots \Rightarrow \Rightarrow R^n \subseteq R^{n-1} \Rightarrow \ldots$. Використовуючи одержане включення для спрощення виразу, отримуємо

$$R_t = R \cup R^2 \cup R^3 \cup \ldots \cup R^n \cup R^{n+1} \cup \ldots = \\ = R \cup R^2 \cup R^3 \cup \ldots \cup R^n \cup \ldots = R$$

b) $(R_t)_t = R_t \cup R_t^2 \cup R_t^3 \cup \ldots \cup R_t^n \cup \ldots = R_t$ (згідно з попереднім пунктом a).

c) Якщо відношення R рефлексивне, то $i_A \subseteq R$ і тоді $i_A \subseteq R \cup R^2 \cup \cup \ldots = R_t$, а це означає, що відношення R_t рефлексивне.

d) Нехай (a, b) — довільний елемент відношення R_t. Це означає, що існують елементи $a_1, \ldots, a_n \in A$ такі, що $a_1 = a$, $a_n = b$ і $a_i R a_{i+1}$, для $i = 1, 2, \ldots, n-1$. В силу симетричності R маємо: $a_{i+1} R a_i$, але тоді $(b, a) \in R_t$. З довільності (a, b) випливає симетричність R_t.

e) Якщо відношення R транзитивне, то візьмемо пару (a, b) і (b, c) із множини $R \cup i_A$. Якщо $(a, b) \in i_A$, то $a = b$ і тоді пара $(a, c) = (b, c)$ належить до $R \cup i_A$. Якщо $(b, c) \in i_A$, то $b = c$ і пара $(a, c) = (a, b)$ належить до $R \cup i_A$. Якщо ні (a, b), ні (b, c) не належать до i_A, то обидві пари належать до R, але тоді $(a, c) \in R \subseteq R \cup i_A$ за властивістю транзитивності відношення R. Отже, у кожному з можливих випадків $(a, c) \in R \cup i_A = R_i$.

f) Для доведення цього пункту, введемо поняття **шляху** в множині A відносно відношення R: будемо говорити, що від елемента a з A до елемента $b \neq a$ з A існує шлях $l = (a = a_1, \ldots, a_k = b)$, якщо

a_1Ra_2, a_2Ra_3, …, $a_{k-1}Ra_k$. З цього означення випливає, що пара $(a, b) \in R_t$ тоді і тільки тоді, коли в множині A існує шлях від елемента a до елемента b. Якщо такий шлях існує, то існує також шлях від a до b, який не проходить двічі через один і той самий елемент, за винятком, коли $a = b$. Такий шлях, який не включає однакових елементів з A, може мати не більше ніж n різних елементів множини A. А це означає, що $(a, b) \in R^k$ для деякого $k \le n$, тобто $R_t = R \cup R^2 \cup \ldots \cup R^n = \bigcup_{k=1}^{n} R^k$. ■

1.4.3. Найменше відношення еквівалентності

Нехай R — деяке бінарне відношення на множині A. Чи існує найменше відношення еквівалентності на множині A, яке включає відношення R? Відповідь на це запитання випливає з такого твердження.

Теорема 10. *Для довільного бінарного відношення R на множині A завжди існує найменше відношення еквівалентності R', яке включає R, і $R' = ((R_i)_s)_t$.*

Доведення. Оскільки відношення R_i рефлексивне, то відношення $((R_i)_s)_t$ теж рефлексивне згідно з попередньою теоремою. За тією ж теоремою відношення $((R_i)_s)_t$ є симетричним і транзитивним. А отже, $R' = ((R_i)_s)_t$ є відношенням еквівалентності.

Нехай R'' — деяке відношення еквівалентності, таке, що $R \subseteq R''$. Тоді $R_i \subseteq R_i'' = R''$, а отже, $(R_i)_s \subseteq (R'')_s = R_s'' = R''$, а звідси випливає — $((R_i)_s)_t \subseteq R_t'' = R''$. Таким чином, $((R_i)_s)_t$ є найменшим відношенням еквівалентності, що включає відношення R. ■

Приклад 1.1.13. Нехай R і R' є відношеннями, визначеними на множині $A = \{1, 2, 3, 4, 7, 9\}$, де

$$R = \{(7, 3), (7,1), (3, 9), (3,1), (2, 2), (4, 2)\},$$
$$R' = \{(4, 3), (4, 9), (4, 1), (3, 3), (4, 2)\}.$$

Тоді

а) рефлексивним замиканням відношення R є відношення

$$R_{i_A} = R \cup i_A = \{(7, 3), (7, 1), (3, 9), (3, 1), (2, 2), (4, 2),$$
$$(1, 1), (3,3), (4, 4), (7, 7), (9, 9)\},$$

а рефлексивним замиканням відношення R' є відношення

$$R'_{i_A} = R' \cup i_A = \{(4, 3), (4, 9), (4, 1), (3, 3), (4, 2), (1, 1), (2, 2), (4, 4), (7, 7), (9, 9)\};$$

b) симетричним замиканням відношення R є відношення

$$R_s = R \cup R^{-1} = \{(7, 3), (7, 1), (3, 9), (3, 1), (2, 2), (4, 2), (3, 7), (1, 7), (9, 3), (1, 3), (2, 4)\},$$

а симетричним замиканням відношення R' є відношення

$$R'_s = R' \cup R'^{-1} = \{(4, 3), (4, 9), (4, 1), (3, 3), (4, 2), (3, 4), (9, 4), (1, 4), (2, 4)\};$$

c) транзитивним замиканням відношення R є відношення

$$R_t = R \cup R^2 \cup \ldots = \{(7, 3), (7, 1), (3, 9), (3, 1), (2, 2), (4, 2), (7, 9)\},$$

а транзитивним замиканням відношення R' є відношення

$$R'_t = R' \cup R'^2 \cup \ldots = R' = \{(4, 3), (4, 9), (4, 1), (3, 3), (4, 2)\};$$

d) найменшим відношенням еквівалентності, що включає відношення R', є відношення

$$R'' = ((R'_{i_A})_s)_t = \{(4, 3), (4, 9), (4, 1), (3, 3), (4, 2), (1, 1), (2, 2), (4, 4), (7, 7), (9, 9), (3, 4), (9, 4), (1, 4), (1, 2), (1, 3), (1, 9), (2, 1), (2, 3), (2, 4), (2, 9), (3, 1), (3, 2), (3, 9), (9, 1), (9, 2), (9, 3)\}.$$

Відношення еквівалентності R'' визначає таке розбиття множини A на класи еквівалентності:

$$A_1 = \{1, 2, 3, 4, 9\}, A_2 = \{7\}. \spadesuit$$

Зауважимо на закінчення, що коли деяке відношення включає своє симетричне, рефлексивне і транзитивне замикання, то воно є відношенням еквівалентності, і навпаки.

1.5. Відображення і операції

Відношення F, задане на множинах $A_1, A_2, \ldots, A_n, B$, називається **функціональним,** якщо для довільного елемента $(a_1, a_2, \ldots, a_n) \in \in A_1 \times A_2 \times \ldots \times A_n$ існує не більше одного елемента b із B такого, що $(a_1, a_2, \ldots, a_n, b) \in F$. Якщо такий елемент b із B існує для деякого $(a_1, a_2, \ldots, a_n)$, то він позначається через $F(a_1, a_2, \ldots, a_n)$ і записується так: $b = F(a_1, a_2, \ldots, a_n)$.

Нехай

$$F^{-1}(b) = \{(a_1, a_2, \ldots, a_n) \in A_1 \times A_2 \times \ldots \times A_n | F(a_1, a_2, \ldots, a_n) = b\}$$

і

$$Dom\,(F) = \bigcup_{b \in B} F^{-1}(b).$$

Очевидно, що для довільного функціонального відношення F, заданого на $A_1, A_2, \ldots, A_n, B$, має місце включення

$$Dom(F) \subseteq A_1 \times A_2 \times \ldots \times A_n.$$

Визначення 15. *Відношення F називається* **повністю визначеним,** *якщо $Dom\,(F) = A_1 \times A_2 \times \ldots \times A_n$ і* **частково визначеним** *або просто* **частковим,** *якщо $Dom\,(F) \subset A_1 \times A_2 \times \ldots \times A_n$.*

Визначення 16. *Відношення F, задане на множинах $A_1, A_2, \ldots, A_n, B$, називається* **відображенням,** *або* **функцією** *із $A_1 \times A_2 \times \ldots \times A_n$ у B (F: $A_1 \times A_2 \times \ldots \times A_n \to B$), якщо F функціональне і повністю визначене. Відношення F називається* **частковим відображенням,** *або* **частковою функцією,** *якщо F функціональне і часткове. Число n називається* **арністю** *функціонального відношення F.*

Якщо F: $A_1 \times A_2 \times \ldots \times A_n \to B$ і існує b із B такий, що $F(a_1, a_2, \ldots, a_n) = b$, то елемент b називають *образом* елемента $(a_1, a_2, \ldots, a_n)$ при відображенні F, а елемент $(a_1, a_2, \ldots, a_n)$ — *прообразом* елемента b. Множину

$$F^{-1}(b) = \{(a_1, a_2, \ldots, a_n) | F(a_1, a_2, \ldots, a_n) = b\},$$

введену раніше, називають *повним прообразом* елемента b в множині $A_1 \times A_2 \times \ldots \times A_n$.

Визначення 17. *Відношення $F : A_1 \times A_2 \times \ldots \times A_n \to B$ називають* **відображенням на** *B тоді і тільки тоді, коли $(\forall b \in B)(F^{-1}(b) \neq \varnothing)$.*

Визначення 18. *Відображення F множини $A_1 \times A_2 \times \ldots \times A_n$ на множину B називається* **взаємно однозначним відображенням** *або* **взаємно однозначною відповідністю** *тоді і тільки тоді, коли обернене до відношення F — відношення F^{-1} — є відображенням B на $A_1 \times A_2 \times \ldots \times A_n$.*

Приклади. 1) Нехай дано множини $X = \{a, b, c, d\}$, $Y = \{2, 3, 4, 5, 6\}$ і функція f, тобто правило, за яким елементам із множини X став-

ляться у відповідність елементи множини Y і яке визначене таким чином:

$$f(a) = 2,\ f(b) = 4,\ f(c) = 6,\ f(d) = 3.$$

Тоді прообразом елемента 4 буде одноелементна множина b, прообразом елемента 5 буде пуста множина $\varnothing$, а множина значень даної функції складається з елементів $\{2, 4, 6, 3\}$.

Функцію f можна представити за допомогою такої діаграми:

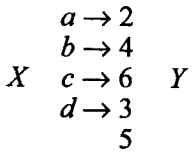


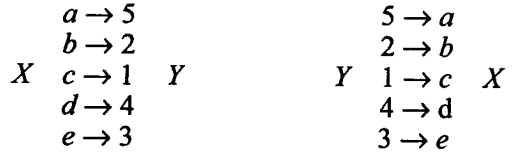
Рис. 1.1.1. Діаграма функції f

2) Нехай $X = \{1, 2, 3, ..., 10\}$, $Y = X$ і функція f є такою:

$$f(1) = 1,\ f(2) = 4,\ f(3) = 9,\ f(4) = 5,\ f(5) = 3,$$
$$f(6) = 3,\ f(7) = 5,\ f(8) = 9,\ f(9) = 4,\ f(10) = 1.$$

Для цієї функції маємо таку множину значень $Y = \{1, 3, 4, 5, 9\}$. Звідси випливає, що дана функція є повністю визначеною, є відображенням у множину Y, а не на множину Y (елементи 2, 6, 7, 8, 10 мають пусті прообрази в X), а отже, не є взаємно однозначним відображенням.

3) Нехай $X = \{a, b, c, d, e\}$, $Y = \{1, 2, 3, 4, 5\}$, розглянемо функцію f, визначену за допомогою такої діаграми:



З цієї діаграми випливає, що $f: X \to Y$ є функцією «на» і, навіть, взаємно однозначною і для цієї функції існує обернена до неї функція $g = f^{-1} : Y \to X$. ♠

Вище була введена операція добутку відношень. Оскільки відображення — це відношення спеціального виду, то з'ясуємо, що собою являє добуток відображень.

Нехай $F_1 : A \to B$, а $F : B \to C$ — деякі відображення. З означення операції добутку відношень маємо: $(a, c) \in F_1 * F \Leftrightarrow$ існує елемент $b \in B$ такий, що $(a, b) \in F_1$ і $(b, c) \in F$, тобто $F_1(a) = b$ і $F(b) = c$, або $F(F_1(a)) = c$ згідно з прийнятими вище позначеннями. Таким чином,

добуток відображень являє собою добре відому операцію суперпозиції функцій.

Покажемо, що операція добутку відображень є асоціативною операцією.

Нехай $F: A \to B$, $F_1 : B \to C$, $F_2 : C \to D$ — довільні відображення. Необхідно показати, що $(F * F_1) * F_2 = F * (F_1 * F_2)$.

Знайдемо, чому дорівнює права частина цього рівняння для довільного елемента a із A: $F * (F_1 * F_2)(a) = F((F_1 * F_2)(a)) = (F_1 * F_2)(F(a)) = F_2(F_1(F(a)))$ і чому дорівнює ліва частина: $(F * F_1) * F_2(a) = F_2((F * F_1)(a)) = F_2(F_1(F(a)))$. Отже, обидві частини рівняння мають один і той самий вираз і через те рівні між собою.

Більш змістовним прикладом відображення на і взаємно однозначного відображення є відображення, яке зв'язує множину A з її фактор-множиною A/R, де R — деяке відношення еквівалентності на A. Відображення $F : A \to A/R$, яке співставляє елементу a із A той клас розбиття, якому належить a, називається **натуральним відображенням** A на A/R.

Між відношеннями еквівалентності, заданими на деякій множині, і відображеннями цієї множини на інші множини існує тісний зв'язок. Дійсно, якщо $F_1 : A \to B$ — відображення на, то йому відповідає цілком визначене відношення еквівалентності R на A: якщо $a, b \in A$, то aRb тоді і тільки тоді, коли $F_1(a) = F_1(b)$. Відношення R називається таким, що відповідає відображенню F_1. Співставляючи кожному елементу x із B його повний прообраз в A, одержуємо відображення $F_2 : B \to A/R$, основну властивість якого дає теорема 11.

Теорема 11. *Якщо F_1 є відображенням множини A на множину B і R — відношення еквівалентності на A, що відповідає F_1, то відображення $F_2 : B \to A/R$ є взаємно однозначним відображенням, причому $F_1 * F_2 = F$, де F — натуральне відображення A на A/R.*

Доведення. Визначимо $F_2: B \to A/R$ так: $F_2(b) = K(b) = \{a \in A | F_1(a) = b\}$. Зрозуміло, що коли $b \neq b'$, то $K(b) \neq K(b')$. Отже, відображення F_2 взаємно однозначне. Далі, нехай $F_1(a) = b$, а $F_2(b) = K(b)$, тоді $a \in K(b)$. Таким чином, добуток $F_1 * F_2$ збігається з натуральним відображенням F. ■

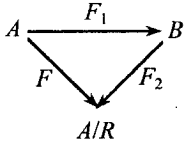


Рис. 1.1.2. Ілюстрація теореми 11

Визначення 19. *Якщо* $F : A^n \to B$, *то* F *називають* n-арною функцією *із* A *в* B, *а якщо при цьому* $B = \{0, 1\}$, *то* F *називається* n-арним предикатом *на множині* A, *а елементи* **0, 1** *відповідно* **хибністю** *і* **істиною.** *Якщо предикат* F *має ту властивість, що для всіх* $(a_1, a_2, \ldots, a_n) \in A$

$$F(a_1, a_2,..., a_n) = 1 \ (F(a_1, a_2,..., a_n) = 0),$$

то предикат F *називається* **тотожно істинним (хибним) на** A.

Між відношеннями і предикатами, заданими на одній і тій самій множині A, існує взаємно однозначна відповідність. Дійсно, нехай F — n-арний предикат на A. Сукупність тих елементів із A^n, для яких $F(a_1, a_2,\ldots, a_n) = 1$ є відношенням на A, яке відповідає предикату F. Навпаки, нехай задане яке-небудь n-арне відношення $R \subseteq A^n$ на A. Покладаючи

$$F(a_1, a_2,..., a_n) = \begin{cases} 1, & \text{якщо } (a_1, a_2,..., a_n) \in R, \\ 0, & \text{якщо } (a_1, a_2,..., a_n) \notin R, \end{cases}$$

отримуємо n-арний предикат, який відповідає відношенню R. Таким чином, n-арні відношення і n-арні предикати на довільній множині знаходяться у взаємно однозначній відповідності, і їх можна не розрізняти.

Якщо F — n-арна функція із A^n в A, то F називають ***n*-арною операцією на** A. При $n = 0$ операція F називається нульарною, значенням якої є фіксований елемент множини A. Операція F називається **частковою,** якщо F — часткова функція.

Нехай $F : A_1 \times A_2 \times ... \times A_n \to B$ і $F^{-1}(b)$ — повний прообраз елемента b в $A_1 \times A_2 \times ... \times A_n$ при відображенні F. Введена вище множина

$$Dom(F) = \bigcup_{b \in B} F^{-1}(b)$$

називається **областю визначення** відображення F, а $Im(F) = \{b | F^{-1}(b) \neq \varnothing\}$ — **областю значень.** Якщо $F : A_1 \times A_2 \times ... \times A_n \to B$ і $F_1 : A_1 \times A_2 \times ... \times A_n \to B$, то $F = F_1 \Leftrightarrow Dom(F) = Dom(F_1)$, $Im(F) = Im(F_1)$ і $F(a_1, a_2,..., a_n) = F_1(a_1, a_2,..., a_n)$ для довільної n-ки $(a_1, a_2,..., a_n) \in A_1 \times A_2 \times ... \times A_n$.

Надалі будемо позначати предикати, функції, операції малими латинськими буквами і якщо Ω — деяка множина предикатів, функцій або операцій, то функція $ar : \Omega \to N$, де N — множина натуральних чисел, називається функцією арності, тобто $ar(\omega) = n$, якщо $\omega \in \Omega$ і має арність n.

1.6. Відношення часткового порядку

Визначення 20. *Бінарне відношення O, визначене на множині A, називається* **частковим порядком** *на A, якщо воно рефлексивне, транзитивне і антисиметричне, тобто якщо для довільних елементів a, b, c із A виконуються властивості:*
a1) aOa $(i_A \subseteq O)$ (*рефлексивність*);
a2) aOb і $bOc \Rightarrow aOc$ $(O^2 \subseteq O)$ (*транзитивність*);
a3) aOb і $bOa \Rightarrow a = b$ $(O \cap O^{-1} \subseteq i_A)$ (*антисиметричність*).

Частковий порядок на множині A, як правило, позначають символом $\leq$, а саму частково упорядковану множину A через $(A, \leq)$. Якщо $a \leq b$ для деяких $a, b \in A$, то говорять, що елементи a і b є такими, що порівнюються між собою і a менше або рівне b, або що a включається в b або рівне b.

Визначення 21. *Транзитивне і іррефлексивне відношення називається відношенням строгого порядку. Це відношення позначають $<$. Транзитивне і рефлексивне відношення називається відношенням квазіпорядку. Це відношення позначають $\preceq$.*

З кожним відношенням часткового порядку $\leq$ пов'язане відношення строгого порядку $<$ (цей порядок називають строгою частиною відношення $\leq$):

$$(a < b) \Leftrightarrow a \leq b \text{ і } a \neq b.$$

Навпаки, з кожним відношенням строгого порядку $<$ пов'язане відношення часткового порядку $\leq$:

$$(a \leq b) \Leftrightarrow a < b \text{ або } a = b.$$

З кожним відношенням квазіпорядку $\preceq$ пов'язані відношення строгого порядку $<$ і еквівалентності $\sim$:

$$(a < b) \Leftrightarrow a \preceq b \text{ і } \neg(b \preceq a);$$

$$(a \sim b) \Leftrightarrow a \preceq b \text{ і } b \preceq a.$$

Довільне відношення квазіпорядку $\preceq$, задане на множині A, індукує відношення часткового порядку $\leq$ на фактор-множині $A/\sim$:

$$([a]_\sim \leq [b]_\sim) \Leftrightarrow a \preceq b.$$

Дійсно, оскільки відношення $\preceq$ транзитивне і рефлексивне, то і відношення $\leq$ теж буде транзитивним і рефлексивним. Покажемо антисиметричність. $[a]_\sim \leq [b]_\sim$ і $([b]_\sim \leq [a]_\sim$ означають, що $a \preceq b$ і $b \preceq a$. Але звідси випливає, що $a \sim b$, тобто, що $[a]_\sim = [b]_\sim$.

Теорема 12. (*Принцип двоїстості*) *Відношення, обернене до відношення часткового порядку, теж буде відношенням часткового порядку.*

Доведення. Нехай $\leq^{-1}$ — відношення, обернене до відношення $\leq$.

б1) Оскільки $i_A \subseteq \leq$, то $i_A = i_A^{-1} \subseteq \leq^{-1}$.

б2) Якщо $\leq * \leq \subseteq \leq$, то згідно з теоремою 3 $\leq^{-1} * \leq^{-1} = (\leq * \leq)^{-1} \subseteq \leq^{-1}$.

б3) Якщо $\leq \cap \leq^{-1} \subseteq i_A$, то $\leq^{-1} \cap \leq \subseteq i_A$ і $\leq^{-1} \cap (\leq^{-1})^{-1} \subseteq i_A$ згідно зі співвідношенням М1 і теоремою 3. ■

Визначення 22. *Відношення часткового порядку $\leq^{-1}$ називається* **двоїстим до** *відношення часткового порядку $\leq$.*.

Відношення $\leq^{-1}$ позначається $\geq$ і $a \leq^{-1} b$ означає $a \geq b$. Якщо $a \leq b$ або $b \leq a$, то a, b називають елементами, що **порівнюються відносно порядку** $\leq$.

З принципу двоїстості випливає, що коли в якому-небудь твердженні про частково упорядковану множину замінити частковий порядок на двоїстий до нього порядок, то одержане твердження теж буде правильним.

Теорема 13. *Довільна підмножина частково упорядкованої множини теж буде частково упорядкованою множиною.*

Доведення пропонується як проста вправа ■

Визначення 23. *Елемент x із множини $(A, \leq)$ називається* **мінімальним (максимальним) елементом** *A, якщо для довільного елемента a із A, що порівнюється з x, має місце нерівність $x \leq a$ ($x \geq a$). Елемент x із A називається* **найбільшим (найменшим),** *якщо $(\forall a \in A)\, x \geq a$ ($x \leq a$).*

Теорема 14. *В довільній частково упорядкованій множині $(A, \leq)$ існує не більше одного найменшого (а відповідно до принципу двоїстості і найбільшого) елемента.*

Доведення. Припустимо, що a і b — два найменші елементи в множині A, тоді $a \leq b$, зважаючи на те, що a найменший елемент, і $b \leq a$, зважаючи на те, що b найменший елемент. Але тоді із антисиметричності відношення $\leq$ випливає, що $a = b$ ■

Якщо довільні два елементи із множини A порівнюються відносно $\leq$, то таке відношення називається **лінійним порядком** на A, а множина A — **лінійно упорядкованою**, або **ланцюгом**.

Зрозуміло, що коли лінійно упорядкована множина A має найбільший (найменший) елемент, то цей елемент буде її єдиним максимальним (мінімальним) елементом.

Визначення 24. *Нехай $(A, \leq)$ — частково упорядкована множина і $a, b \in A$. Вважається, що елемент b домінує над елементом a, якщо $b > a$ і ні для якого елемента x із A невірно, що $b > x > a$.*

Наступна проста теорема показує, що відношення часткового порядку $\leq$ можна однозначно відновити за відношенням домінування в довільній скінченній частково упорядкованій множині.

Теорема 15. *Нехай $a < b$ в скінченній частково упорядкованій множині $(A, \leq)$. Тоді в $(A, \leq)$ існує хоча б один ланцюг $a < x_1 < x_2 < \ldots < x_n < b$, в якому кожний x_i домінує над x_{i-1}, $i = 1, 2, \ldots, n$.*

Доведення індукцією за числом n елементів y, які задовольняють умові $a < y < b$ (див. нижче метод трансфінітної індукції).

(*База індукції* $n = 0$). У цьому випадку b домінує над a за означенням.

(*Крок індукції*) Припустимо, що теорема правильна для всіх $m < n$. Розглянемо випадок $n = m$, де $n > 0$. Оскільки $n > 0$, то існує такий елемент $c \in A$, що $a < c < b$ і число елементів y, z, що задовольняють умовам $a < y < c$ і $c < z < b$ не перевищує $n - 1$. За припущенням індукції існують скінченні ланцюги, які зв'язують a і c і c і b, середні елементи яких знаходяться у відношенні домінування. З'єднавши ці два ланцюги, одержуємо шуканий ланцюг ■

Приклади частково упорядкованих множин

в) Булеан $B(A)$, тобто множина всіх підмножин деякої множини A з відношенням теоретико-множинного включення $\subseteq$ як відношенням часткового порядку.

г) Множина N^+ з відношенням $n \leq n_1 \Leftrightarrow n_1$ ділиться без залишку на n ♠

Приклади лінійно упорядкованих множин

1. Множини N, N^+, Z, Q і множина D дійсних чисел з їх звичайним порядком. Множину N часто називають натуральним рядом, а довільну її підмножину вигляду $\{0, 1, 2, \ldots, n\}$ — початковим відрізком, або просто відрізком натурального ряду.

2. Множина точок числової осі (прямої).

3. Нехай маємо скінченний алфавіт $X = \{x_1, x_2, \ldots, x_n\}$, літери якого лінійно упорядковані:

$$x_i < x_j \Leftrightarrow i < j.$$

Лексикографічним порядком на множині $F(X)$ називається відношення $\leq$, для якого

$$p = x_{11}x_{12} \ldots x_{1k} \leq x_{21}x_{22}\ldots x_{2r} = q$$

тоді і тільки тоді, коли виконується одна із двох умов:

a) слово p є початком слова q;

b) існує таке натуральне число j, що $x_{1j} < x_{2j}$ і для всіх $i < j$ справедливі рівності $x_{1i} = x_{2i}$.

Згідно з лексикографічним порядком упорядковані слова в словниках, якщо прийнятий порядок літер у деякому алфавіті розуміти як лінійний порядок для його літер. ♠

Лінійно упорядкована множина A називається **повністю упорядкованою,** якщо довільна її непуста підмножина B має найменший елемент. З поняттям повністю упорядкованої множини пов'язаний один з основних постулатів теорії множин.

АКСІОМА ПОВНОЇ УПОРЯДКОВАНОСТІ

Довільну непусту множину можна повністю упорядкувати.

Зауважимо, що ця аксіома логічно еквівалентна другій аксіомі теорії множин — аксіомі вибору [20, 21, 25].

АКСІОМА ВИБОРУ

Якщо дано множину A, то існує функція f, яка ставить у відповідність кожній непустій підмножині B із множини A один визначений елемент $f(B)$ із множини B.

Логічну еквівалентність аксіом слід розуміти так: коли одну із них вибрати за аксіому, то другу можна строго довести як теорему, і навпаки. Так, якщо приймається аксіома вибору, то аксіома повної упорядкованості стає твердженням, яке в теорії множин відоме як теорема Цермело.

Завдяки аксіомі повної упорядкованості багато властивостей повністю упорядкованих множин можна доводити методом трансфінітної індукції.

Метод трансфінітної індукції. *Нехай e — найменший елемент повністю упорядкованої множини A і $P(x)$ — деяка властивість елемента $x \in A$. Тоді, якщо із істинності $P(e)$ і $P(x)$ для всіх $x < a$ випливає істинність $P(a)$, то $P(x)$ істинно для всіх x із A.*

Доведення. Припустимо супротивне, тобто припустимо, що існує така непуста підмножина A' елементів із A, що $P(a)$ хибне на її елементах за виконання умов теореми. Нехай a — мінімальний елемент в A'. Оскільки $P(e)$ істинне, то $a \neq e$ і $a > e$. Із умов теореми випливає, що $P(x)$ істинне для всіх $x < a$, але тоді з цих самих умов має випливати істинність і $P(a)$, а це суперечить нашому припущенню. ■

Зауважимо, що множина натуральних чисел N є повністю упорядкованою множиною і тому її можна взяти за множину A, яка фігурує в попередній теоремі. Якщо повністю упорядкованою множиною є множина натуральних чисел N, то метод трансфінітної індукції називається методом математичної індукції. Розглянемо приклад застосування методу математичної індукції до доведення тверджень.

Приклад використання методу математичної індукції

Довести, що сума трьох послідовних кубів, відмінних від нуля натуральних чисел, ділиться на 9.

Доведення. (База індукції $n = 1$). При $n = 1$ маємо $1^3 + 2^3 + 3^3 = 36$ і оскільки 36 ділиться на 9, то базис індукції має місце.

(Крок індукції) Щоб довести крок індукції потрібно припустити, що дане твердження є правильним для n і довести, що воно буде правильним і для $n + 1$. Отже, припустимо, що $n^3 + (n + 1)^3 + (n + 2)^3$ ділиться на 9. Тоді

$$(n + 1)^3 + (n + 2)^3 + (n + 3)^3 = (n + 1)^3 + (n + 2)^3 + n^3 + 9n^2 + 27n + 27 = \\ = n^3 + (n + 1)^3 + (n + 2)^3 + 9(n^2 + 3n + 3).$$

Але всі доданки в одержаній сумі діляться на 9, оскільки сума перших трьох ділиться на 9 за припущенням індукції, а останній доданок кратний 9. Отже, вся сума теж кратна 9. ♠

Метод індукції дає можливість не тільки доводити твердження, але й виконувати побудову по індукції і давати означення по індукції.

Дійсно, нехай A — повністю упорядкована множина і нехай на цій множині визначена функція $f(x)$, яка ставить у відповідність кожному елементу x із A деякий елемент множини B. Припустимо також, що $f(x)$ повинна задовольняти деяким *рекурентним співвідношенням*, тобто співвідношенням, які однозначно визначають для довільного $a \in A$ значення $f(a)$ за значеннями $f(b)$ для всіх $b < a$.

Метод побудови по індукції. *Існує єдина функція $f(x)$, яка визначена на всій множині A, задовольняє вказаним рекурентним співвідношенням і приймає задані значення на мінімальному елементі множини A.*

Доведення. Покажемо спочатку єдиність такої функції. Припустимо, що існує дві різні функції $f(x)$ і $g(x)$ на множині A, які задовольняють нашим умовам. Нехай існує непуста підмножина елементів x із A, для яких $f(x) \neq g(x)$. Оскільки A — повністю упорядкована, то ця підмножина має мінімальний елемент a. Цей елемент не може бути мінімальним для всієї множини A, тому що тоді, за умовою, на цьому елементі $f(a)$ і $g(a)$ збігались би. Отже, існує $b < a$ такий, що $f(b) = g(b)$. За умовою теореми, рекурентні співвідношення однозначно визначають значення наших функцій для $x = a$ по їх значеннях для всіх $b < a$, а це означає, що $f(a) = g(a)$. Одержана суперечність доводить єдиність $f(x)$.

Доведемо тепер існування функції. Припустимо, що на мінімальному елементі множини A значення шуканої функції уже задано. Позначимо через P таку властивість: елемент $a \in A$ задовольняє властивості P, якщо на множині C усіх таких x, що $x \leq a$, може бути визначена функція $f_a(x)$, яка задовольняє рекурентним співвідношенням і приймає задане значення на мінімальному елементі множини A.

В силу припущення P істинне на мінімальному елементі $e \in A$. Далі, якщо елементи b і c задовольняють властивості P і $b < a$, то в силу доведеної вище єдиності шуканої функції не на множині A, а на множині $B = \{x \in A | x \leq b\}$, маємо $f_b(x) = f_a(x)$.

Звідси випливає, що коли всі елементи b, строго менші від елемента a, задовольняють властивості P, то і сам елемент a задовольняє цій властивості. Одержуємо функцію $f_a(x)$, яка задовольняє всі вимоги, якщо для довільного елемента $b < a$ покласти $f_a(b) = f_b(b)$, а за $f_a(a)$ взяти те значення, яке однозначно визначається рекурентними співвідношеннями.

На основі методу трансфінітної індукції можна стверджувати, що для всіх a із A істинно $P(a)$. Припускаючи, що тепер $(\forall a \in A)\, f_a(a) = f(a)$, визначаємо функцію $f(x)$, яка має всі необхідні властивості. ■

Зауважимо, що метод трансфінітної індукції, як і метод побудови по індукції, можна застосовувати і до частково упорядкованих множин. Обмежимося лише формулюваннями цих фактів.

Умова індуктивності. *Усі елементи частково упорядкованої множини A задовольняють властивості P, якщо*:

1) *всі мінімальні елементи із множини A задовольняють властивості P (у тому випадку, коли вони існують)*;

2) *із того, що $P(x)$ істинне для всіх $x < a$, де $x, a \in A$, випливає істинність $P(a)$.*

Побудова по індукції. *Існує єдина функція $f(x)$, яка визначена на всій множині A, задовольняє вказаним рекурентним співвідношен-*

ням і приймає задані значення на всіх мінімальних елементах множини A [21, 25].

На завершення цього підрозділу розглянемо кілька прикладів відображень.

1.7. Приклади відображень

1.7.1. Потужність множини

Розглянемо питання про те, як порівнюються між собою множини за кількістю своїх елементів. Нехай A і B — довільні дві множини.

Визначення 25. *Множини A і B називаються* **рівнопотужними,** *якщо між їх елементами існує взаємно однозначна відповідність.*

Очевидно, що відношення рівнопотужності є відношенням еквівалентності і тому рівнопотужні множини часто називають **еквівалентними.**

Визначення 26. **Потужністю** *або* **кардинальним числом** *множини A називається клас еквівалентності, якому належить A, за відношенням рівнопотужності.*[1]

Якщо A і B скінченні множини, то їх рівнопотужність означає, що вони мають одне й те саме число елементів. При цьому пустій множині $\varnothing$ відповідає число нуль, а скінченній множині, яка складається із n елементів — число n. Право на таку відповідність нам дає наступна теорема.

Теорема 16. (*Основна теорема про скінченні множини*). *Довільна скінченна множина не може бути еквівалентною якій-небудь своїй власній надмножині.*

Доведення. Припустимо, що існує деяка скінченна надмножина B скінченної множини A, яка еквівалентна A, тобто існує взаємно однозначна відповідність $f : A \to B$. Нехай $a_1, a_2, \ldots, a_n$ — елементи множини A, а $f(a_1), f(a_2), \ldots, f(a_n)$ — їх образи. Серед образів мають

[1] Необхідно зауважити, що це визначення приводить до деяких труднощів логічного характеру, які за допомогою певних уточнень можна обійти [16, 20, 36]. Одна з них полягає в тому, що відношення рівнопотужності визначається на «множині всіх множин», а поняття «множина всіх множин» вимагає уточнення.

бути всі елементи множини A і ще хоча б один елемент, який позначимо a_{n+1}.

Для $n = 1$ суперечність очевидна, оскільки єдиний елемент a_1 не може мати два різних між собою образи — a_1, a_2.

Нехай неможливість існування відображення f із вказаними властивостями доведена для множини A з $n - 1$ елементами. Доведемо її для множини A з n елементами.

Можна вважати, що $f(a_n) = a_{n+1}$, тому що коли це не так, тобто $f(a_n) = a'$ і $a' \neq a_{n+1}$, то a_{n+1} має інший прообраз — $a_i : f(a_i) = a_{n+1}$. Тоді можна побудувати інше, відмінне від f відображення, яке буде співставляти елементу a_n елемент a_{n+1}, а елементу a_i — елемент a', а всім іншим не буде відрізнятися від f.

Підмножина $A' = \{a_1, a_2, \ldots, a_{n-1}\}$ відображається функцією f на деяку множину $f(A')$, яка будується із $f(A) = B$ за допомогою відкидання елемента $f(a_n) = a_{n+1}$.

Множина $f(A')$ включає елементи $a_1, \ldots, a_n$ і, отже, є власною надмножиною множини A' і разом з тим її взаємно однозначним образом. З огляду на припущення індукції це неможливо. ■

З цієї теореми випливає, що скінченна множина ніколи не може бути еквівалентною двом різним відрізкам натурального ряду, тому що в протилежному випадку ці відрізки були б рівнопотужними, і при цьому один із них мав би включати другий. Таким чином, довільна скінченна множина A еквівалентна одному і тільки одному відрізку $\{0, 1, 2, \ldots, n\}$ натурального ряду. Однозначно визначене число m — число елементів множини A — служить мірою потужності цієї множини.

Теорема 17. *Якщо $A_1, A_2, \ldots, A_n$ — скінченні множини, то*

$$|A_1 \times A_2 \times \ldots \times A_n| = |A_1|\,|A_2| \ldots |A_n| .$$

Доведення. Спочатку покажемо, що $|A_1 \times A_2| = |A_1|\,|A_2|$. Нехай $A_1 = \{a_{11}, a_{12}, \ldots, a_{1n}\}$, $A_2 = \{a_{21}, a_{22}, \ldots, a_{2k}\}$ і $B(a_{1i}) = \{(a_{1i}, a_{2j})\}$, де a_{1i} — деякий фіксований елемент множини A_1. Тоді $|A_2| = |B(a_{1i})|$, оскільки ці множини еквівалентні (елементу (a_{1i}, a_{2j}) відповідає елемент a_{2j}, $j = 1, 2, \ldots, k$). Звідси маємо, що

$$|A_1 \times A_2| = |B(a_{11})| + |B(a_{12})| + \ldots + |B(a_{1n})| = |A_1|\,|A_2|,$$

оскільки множини $B(a_{1i})$ і $B(a_{1j})$ попарно не перетинаються при $i \neq j$.

Звертаючи увагу на те, що множини $A_1 \times (A_2 \times \ldots \times A_n)$ і $A_1 \times A_2 \times \ldots \times A_n$ еквівалентні (елементові $[a_1, (a_2, \ldots, a_n)]$ відповідає елемент $(a_1, a_2, \ldots, a_n)$), можемо записати

$$|A_1 \times A_2 \times \ldots \times A_n| = |A_1|\,(|A_2 \times \ldots \times A_n|) = |A_1|\,|A_2|\,(|A_3 \times \ldots \times A_n|) = \ldots = |A_1|\,|A_2| \ldots |A_n|. \blacksquare$$

Визначення 27. *Множина, еквівалентна множині натуральних чисел, називається* **зліченною множиною.**

Безпосередньо із означення зліченної множини випливає

Теорема 18. *Довільна нескінченна підмножина зліченної множини сама зліченна.*

Дійсно, перерахунок елементів підмножини B зліченної множини A можна виконати в порядку їх слідування у множині A. ■

У випадку нескінченних множин аналог теореми 16 не має місця, як показує теорема 19.

Теорема 19. *Об'єднання скінченної або зліченної множини зліченних множин є множиною зліченною* [9].

Доведення. Розглянемо спочатку випадок скінченного числа зліченних множин. Нехай $A_1,..., A_k$ — ці множини і $a_{ij} \in A_i$ — їх елементи ($i = 1, 2,..., k$). Розглянемо послідовність

$$a_{11}, \ldots, a_{k1}, \ldots, a_{12}, \ldots, a_{k2}, \ldots, a_{1n}, \ldots, a_{kn}, \ldots .$$

Таку послідовність можна перелічити, і якщо деякий елемент при переліку вже траплявся раніше і одержав номер, то він пропускається надалі. Отже, множина $A = \bigcup_{i=1}^{k} A_i$ зліченна.

Нехай маємо зліченну множину зліченних множин $A_1, A_2, \ldots, A_n,...$, де $A_i = \{a_{i1}, a_{i2}, \ldots\}$. Існує лише скінченне число елементів a_{ik}, для яких $i + k = 2$, аналогічно існує лише скінченне число елементів a_{ik}, для яких $i + k = 3$ і т. д. Перенумеруємо спочатку всі елементи, для яких $i + k = 2$ (наприклад, за зростанням значення i), а потім (за допомогою інших чисел) — елементи, для яких $i + k = 3$ і т. д. При цьому кожний елемент a_{ik} одержить деякий номер, і різні елементи будуть мати різні номери. Звідси випливає справедливість теореми. ■

Наслідок 1. *Множина Z всіх цілих чисел зліченна.*

Дійсно, $Z = N \cup N^-$, де $N^- = \{-1, -2,..., -n,...\}$. ■

Довести цей наслідок можна шляхом побудови явного взаємно однозначного відображення між елементами множин Z і N. Дійсно, розглянемо функцію $f: Z \to N$, яка визначається таким чином:

$$f(n) = \begin{cases} 0, & \text{якщо } n = 0, \\ 2n, & \text{якщо } n > 0, \\ 2n+1, & \text{якщо } n < 0. \end{cases}$$

Ця функція генерує такі пари: ..., (–3, 7), (–2, 5), (–1, 3), (0, 0), (1, 2), (2, 4), (3, 6),.... Очевидно, що коли $m \neq n$, то $(m, f(m)) \neq (n, f(n))$, тобто побудована функція є взаємно однозначним відображенням. А звідси отримуємо, що множини Z і N еквівалентні.

Наслідок 2. *Множина Q всіх раціональних чисел зліченна.*

Множина раціональних чисел — це об'єднання зліченної сукупності зліченних множин вигляду

$$RC = Z \cup R_2 \cup R_3 \cup \ldots \cup R_n \cup \ldots,$$

де $R_n = \{m/n | m \in Z, n = 2, 3, \ldots\}$. ■

Наслідок 3. *Декартовий добуток $A \times B$ зліченних множин є множина зліченна.*

Дійсно, якщо $A = \{a_1, \ldots, a_n, \ldots\}$ і $B = \{b_1, \ldots, b_n, \ldots\}$, то множину всіх елементів множини $A \times B$ можна представити як об'єднання зліченної сукупності зліченних множин вигляду

$$\begin{aligned} A_1 &= \{(a_1, b_1), (a_1, b_2), \ldots, (a_1, b_n), \ldots\}, \\ A_2 &= \{(a_2, b_1), (a_2, b_2), \ldots, (a_2, b_n), \ldots\}, \\ &\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ A_n &= \{(a_n, b_1), (a_n, b_2), \ldots, (a_n, b_n), \ldots\}, \\ &\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots . \end{aligned}$$

На підставі теореми 19 заключаємо, що множина $A \times B$ зліченна. ■

Наслідок 4. *Декартовий добуток $A_1 \times A_2 \times \ldots \times A_n$ зліченних множин $A_1, A_2, \ldots, A_n$ є множина зліченна для довільного скінченного $n \in N$.*

Далі будемо мати справу або зі скінченними або нескінченними зліченними множинами. Але, для повноти, зауважимо, що існують множини, елементи яких перелічити неможливо. Такі множини логічно назвати **незліченними.** Має місце важлива теорема.

Теорема 20 (*Теорема Кантора*). *Множина всіх дійсних чисел з інтервалу* (0, 1) *незліченна* [9].

Доведення цієї теореми ґрунтується на **діагональному методі Кантора.**

З курсу шкільної математики відомо, що кожному дійсному числу з інтервалу (0, 1) можна однозначно співставити правильний нескінченний десятковий дріб $0, a_1 a_2 \ldots a_n \ldots$, який має нескінченно багато цифр.

Припустимо, що теорема хибна і множина дійсних чисел з інтервалу (0,1) зліченна. Тоді для цієї множини повинен існувати перелік

$$x_1 = 0, a_{11}a_{12} \ldots a_{1n}\ldots,$$
$$x_2 = 0, a_{21}a_{22} \ldots a_{2n}\ldots,$$
$$\ldots\ldots\ldots\ldots$$
$$x_n = 0, a_{n1}a_{n2} \ldots a_{nn}\ldots,$$
$$\ldots\ldots\ldots\ldots\ldots\ldots .$$

Прямуючи по діагоналі (тобто по елементах a_{11}, a_{22}, …, a_{nn},...), побудуємо новий дріб 0, $a'_{11}a'_{22}...a'_{nn}...$, такий що $a'_{ii} = 1$, якщо $a_{ii} \neq 1$, і $a'_{ii} = 2$, якщо $a_{ii} = 1$, для довільного $i = 1, 2,\ldots$. Побудований таким чином дріб не входить до розглянутого переліку, оскільки відрізняється від кожного елемента x_i числом a_{ii}, яке розташоване на діагоналі. Отже, переліку всіх чисел з інтервалу (0, 1) не існує і припущення про зліченність цієї множини хибне. ■

Наслідок 5. *Якщо A — незліченна множина і $B \subseteq A$ — деяка зліченна підмножина множини A, то множина $C = A \setminus B$ незліченна.*

Дійсно, якби множина C була зліченною, то згідно з теоремою 19 була б зліченною і множина $A = C \cup B$. ■

Наслідок 6. *Множина IR усіх ірраціональних чисел незліченна.*

Визначення 28. *Множина, еквівалентна множині дійсних чисел з інтервалу* (0, 1), *називається* **множиною потужності континууму.**

Теорема 21 (*Теорема про булеан*). *Булеан $B(U)$ довільної зліченної множини U — незліченна множина.*

Доведення. Не обмежуючи загальності, візьмемо $U = N$. Співставимо кожній підмножині $N_i \subseteq N$ ланцюг, який складається із нулів і одиниць, таким чином: елемент a_{ij}, який стоїть на i-му місці в множині N_j, дорівнює нулю, якщо $a_{ij} \in N_j$, і 1, якщо $a_{ij} \notin N_j$. Очевидно, що така відповідність буде взаємно однозначною. Використовуючи діагональний метод Кантора, побудуємо новий ланцюг

$$a'_{11}a'_{22}...a'_{nn}...,$$

де

$$a'_{ii} = \begin{cases} 1, & \text{якщо } a_{ii} = 0, \\ 0, & \text{якщо } a_{ii} = 1. \end{cases}$$

$i = 1, 2,..., n,...$. Зрозуміло, що такий ланцюг не збігається ні з яким ланцюгом для множини N_j і відповідає деякій підмножині $N' \subseteq N$,

яка не збігається ні з однією з множин N_j, $j = 1, 2,..., n,...$. Отже, елементи $B(U)$ неможливо перелічити. ■

За встановлення еквівалентності двох множин часто буває корисна теорема Кантора-Бернштейна.

Теорема 22. *Якщо множина A еквівалентна деякій підмножині B' множини B і множина B еквівалентна деякій підмножині A' множини A, то множини A і B рівнопотужні.*

1.7.2. Матриці

Розглянемо ще один цікавий приклад відображення. Нехай $p, q \in N^+$, N_p і N_q означають множини чисел $\{1, 2, ..., p\}$ і $\{1, 2,..., q\}$ відповідно, а S — довільна множина чисел.

Визначення 29. *Відображення $A : N_p \times N_q \to S$, де $p, q \in N^+$, називається прямокутною* **матрицею** *над множиною чисел S. Образ $A(i, j)$ часто позначають a_{ij} і називають елементом матриці A, а саму матрицю A задають за допомогою таблиці елементів-образів із множини S, які відповідають відображенню A, тобто*

$$A = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1q} \\ a_{21} & a_{22} & \dots & a_{2q} \\ \dots & \dots & \dots & \dots \\ a_{p1} & a_{p2} & \dots & a_{pq} \end{pmatrix}.$$

При цьому стверджують, що матриця A складається із p рядків і q стовпчиків і має розмірність $p \times q$. Якщо $p = q$, то матриця A називається **квадратною,** *а коли елементи квадратної матриці такі, що $a_{ij} = 0$ при $i \neq j$ і $a_{ij} \neq 0$ при $i = j$, то матриця A називається* **діагональною.** *Діагональна матриця називається* одиничною, *якщо $a_{ii} = 1$.*

Матриці є одним із зручних засобів задання бінарних відношень на скінченних множинах.

Нехай R — бінарне відношення, задане на скінченних множинах $A = \{a_1, a_2,..., a_p\}$ і $B = \{b_1, b_2, ..., b_q\}$. Розглянемо матрицю $A(R)$: $N_p \times N_q \to \{0, 1\}$, яка зв'язана з відношенням R (за допомогою відображення A) таким чином:

$$a_{ij} = \begin{cases} 1, & \text{якщо } (a_i, b_j) \in R, \\ 0, & \text{якщо } (a_i, b_j) \notin R. \end{cases}$$

Наприклад, якщо $A = \{a_1, a_2, a_3, a_4\}$, $B = \{b_1, b_2, b_3\}$ і $R = \{(a_1, b_2), (a_1, b_3), (a_2, b_1), (a_3, b_1), (a_4, b_2)\}$, то матриця $A(R)$: $A \times B \to \{0, 1\}$ має вигляд:

$$A(R) = \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}.$$

Зауважимо, що в окремому випадку, коли R — бінарне відношення, задане на скінченній множині A, то йому відповідає квадратна матриця. Зокрема,

а) якщо $R = i_A$ — відношення тотожності на A, то

$$A(i_A) = \begin{pmatrix} 1 & 0 & \dots & 0 & 0 \\ 0 & 1 & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & 1 & 0 \\ 0 & 0 & \dots & 0 & 1 \end{pmatrix}.$$

одинична діагональна матриця (звідки і назва відношення). Звідси легко знайти рефлексивне замикання деякого заданого відношення R на множині A. Для цього в матриці $A(R)$ необхідно скрізь поставити на діагоналі 1 замість 0.

б) якщо R — симетричне відношення, то матриця $A(R)$ буде, очевидно, симетричною. Дійсно, в силу симетричності відношення R, якщо $(a_i, b_j) \in R$ ($d_{ij} = 1$), то $(b_j, a_i) \in R$ ($a_{ji} = 1$). Приймаючи до уваги цю обставину, легко побудувати за матрицею відношення R, матрицю його симетричного замикання $A(R_s)$. Для цього необхідно в $A(R)$ замінити 0 на 1 на відповідних місцях з урахуванням симетрії. Наприклад, якщо $A = \{a_1, a_2, a_3, a_4\}$ і $R = \{(a_1, a_3), (a_1, a_4), (a_2, a_2), (a_2, a_4), (a_3, a_4))\}$, то

$$A(R) = \begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \quad A(R_s) = \begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}. \ \spadesuit$$

Контрольні питання

1) Що таке множина?
2) Які існують способи задання множин?
3) Які основні операції виконуються над множинами?
4) Що таке булеан деякої множини U?
5) Що таке
   a) декартовий добуток множин A, B, C, E;
   b) декартовий степінь деякої множини A;
   c) бінарне відношення, задане на множині A?
6) Назвіть основні властивості бінарних відношень. Яке відношення називається
   a) рефлексивним; b) симетричним;
   c) транзитивним; d) антисиметричним?
7) Чи будуть відношеннями еквівалентності такі відношення
   a) $R_1 = R \cap T$,
   b) $R_1 = R * T$,
   c) $R_1 = R^{-1}$,

якщо R і T — відношення еквівалентності?

8) Чи буде бінарне відношення R вигляду

$$(a, b)\, R(c, d) \Leftrightarrow a + d = b + c,$$

де $a, b, c, d \in N^+$, відношенням еквівалентності?

9) Чи буде відношенням часткового порядку відношення, обернене до відношення часткового порядку?

10) Якою буде матриця
   a) рефлексивного, б) симетричного, в) антисиметричного відношення, заданого на деякій непустій скінченній множині?

11) Чи будуть еквівалентними множини {a, b, c, d} і {a, b, c, d, d}? Доведіть, що множини парних і непарних натуральних чисел еквівалентні.

Задачі і вправи

1) Довести, що умови $A \subseteq B$, $A \cap B = A$, $A \cup B = B$ еквівалентні між собою.

2. Довести, що $A \cap B \subseteq A \subseteq A \cup B$, $A \cap B \subseteq B \subseteq A \cup B$, $A \setminus B \subseteq A$.

3. Довести, що (а) $\varnothing \neq \{\varnothing\}$, (б) $\{\{a\}, \{b, c\}\} \neq \{a, b, c\}$.

4. Пояснити, чому (а) $3 \in \{1, 2, 3, 4\}$, (б) $\{1, 2\} \notin \{\{1, 2, 3\}, \{2, 3\}, 1, 2\}$.

5. Описати словами кожну з множин:

a) $\{x \in N \mid x$ ділиться на 2 і ділиться на $3\}$,

b) $\{x \mid x \in A\, x \in B\}$,

c) $\{x \mid x \in A$ і $x \notin B\}$

d) $\{(x, y) \in D^2 \mid x^2 + y^2 = 1\}$,

e) $\{(x, y) \in D^2 \mid y = 2x$ і $y = 3x\}$.

6. Дано алфавіт $X = \{a, b, c\}$. Виписати

a) перших десять слів найменшої довжини множини $F(X)$;

b) усі префікси і всі суфікси слова $p = aabbca$;

c) усі підслова слова $p = aabbca$;

d) усі входження слова $p = ab$ в слово $q = abbccaabb$;

e) довести, що операція конкатенації слів у множині $F(X)$ не є комутативною.

7. Яку властивість мають слова, що представляють парні й непарні натуральні числа в двійковій системі числення?

8. Довести, що відображення $f: N \to F(\{0, 1\})$, яке ставить у відповідність натуральному числу n його запис у двійковій системі числення, є взаємно однозначним відображенням.

9. Нехай універсальною множиною U служить множина натуральних чисел N, тобто $U = N$, а

$A = \{x \in N \mid$ для деякого $y \in N^+\, x = 2y\}$,

$B = \{x \in N \mid$ для деякого $y \in N^+\, x = 2y - 1\}$,

$C = \{x \in N \mid x < 10\}$.

Побудувати або описати словами множини A', $(A \cup B)'$, C', $A \backslash C'$, $C \setminus (A \cup B)$.

10. Знайдіть множини

$$\varnothing \cup \{\varnothing\},\ \{\varnothing\} \cup \{\varnothing\},\ \{\varnothing, \{\varnothing\}\} \backslash \varnothing,\ \{\varnothing, \{\varnothing\}\} \backslash \{\varnothing\},\ \{\varnothing, \{\varnothing\}\} \backslash \{\{\varnothing\}\}.$$

11. Довести, що множина всіх коренів многочлена $F(x) = f_1(x) \cdot f_2(x)$ — це об'єднання множин коренів многочленів $f_1(x)$ і $f_2(x)$.

12. Довести закони М1—М5 алгебри множин (див. теорему 1).

13. Довести тотожності:

$A \cup A' = U$, $A \setminus (B \cup C) = (A \setminus B) \cap (A \setminus C)$, $A \setminus (B \cup C) = (A \setminus B) \setminus C$,

$A \setminus (B \cap C) = (A \setminus B) \cup (A \setminus C)$,

$A \setminus (B \setminus C) = (A \setminus B) \cup (A \cap C)$,

$(A \setminus B) \setminus C = (A \setminus C) \setminus (B \setminus C)$,

$A \cap B = A \setminus (A \setminus B)$, $A \cup B = A \cup (B \setminus A)$,

$(A' \cup B) \cap A = A \cap B$, $A \cap (B \setminus A) = \varnothing$,

$(A \cup B) \setminus C = (A \setminus C) \cup (B \setminus C)$,

$A \cup B \subseteq C \Leftrightarrow A \subseteq C$ і $B \subseteq C$,

$A \subseteq (B \cap C) \Leftrightarrow A \subseteq B$ і $A \subseteq C$,

$A \cap (B \setminus C) = (A \cap B) \setminus (A \cap C) = (A \cap B) \setminus C$,

$(A \cap B) \cup (A \cap B') = (A \cup B) \cap (A \cup B') = A$,

$A \subseteq B \Leftrightarrow A \cup B = B \Leftrightarrow A \cap B = A \Leftrightarrow A \setminus B = \varnothing \Leftrightarrow A' \cup B = U$,

де U — універсальна множина, а A' — доповнення множини A в U.

14. Чи існують такі множини A, B, C, що $A \cap B \neq \varnothing$, $A \cap C \neq \varnothing$, $(A \cap B) \setminus C = \varnothing$?

15. Знайти всі підмножини множин: $\varnothing$, $\{\varnothing\}$, $\{x\}$, $\{1, 2\}$.

16. Довести, що множина A, яка складається із n елементів, має 2^n підмножин.

17. Що являє собою множина $D \times D$, якщо D — множина дійсних чисел?

18. Довести, що $A = (B \cup C) \Leftrightarrow (A \cap B') = C$.

19. Які з наведених нижче тверджень справедливі для довільних множин A, B, C:

$$A \subseteq B \text{ і } B \subseteq C \Rightarrow A \subseteq C,$$
$$A \neq B \text{ і } B \neq C \Rightarrow A \neq C,$$
$$A \subseteq (B \cup C)' \text{ і } B \subseteq (A \cup C)' \Rightarrow B = \varnothing.$$

20. Нехай U — універсальна множина і $A \subseteq U$. Довести, що $U \subseteq A \Rightarrow A = U$.

21. Довести, що $A = B' \Leftrightarrow A \cap B = \varnothing$ і $A \cup B = U$, де $A, B \subseteq U$.

22. Довести, що коли $A_1 \subseteq A_2 \subseteq \ldots \subseteq A_n \subseteq A_1$, то $A_1 = A_2 = \ldots = A_n$ для довільних множин $A_1, A_2, \ldots, A_n$.

23. Довести, що

$A \div B = B \div A$, $A \div (B \div C) = (A \div B) \div C$,

$A = B \Leftrightarrow A \div B = \varnothing$,

$A \cap (B \div C) = (A \cap B) \div (A \cap C)$,

$A \div (A \div B) = B$,

$A \cup B = A \div B \div (A \cap B)$,

$A \setminus B = A \div (A \cap B)$,

$A \div \varnothing = A$, $A \div A = \varnothing$, $A \div U = A'$.

24. Довести, що

$A \cup B = A \cap B \Rightarrow A = B$,

$(A \cap B) \cup C = A \cap (B \cup C) \Leftrightarrow C \subseteq A$,

$A \subseteq B \Rightarrow A \cup C \subseteq B \cup C$, $A \subseteq B \Rightarrow A \cap C \subseteq B \cap C$,

$A \subseteq B \Rightarrow A \setminus C \subseteq B \setminus C$, $A \subseteq B \Rightarrow C \setminus B \subseteq C \setminus A$, $A \subseteq B \Rightarrow B' \subseteq A'$,

$(A \cap B) \cup C = A \cap (B \cup C) \Leftrightarrow C \subseteq A$.

25. Довести, що коли A, B, C, D непусті множини, то

$A \subseteq B$ і $C \subseteq B \Leftrightarrow A \times C \subseteq B \times D$,

$A = B$ і $C = D \Leftrightarrow A \times C = B \times D$.

26. Довести, що $B(A \cap C) = B(A) \cap B(C)$, $B(\cap A_i) = \cup B(A_i)$, $B(\cup A_i) = \{\cup C_i | C_i \in B(A_i)\}$, де i пробігає деяку множину цілих чисел I.

27. Довести, що

$$\bigcap_{i=1}^{n}(B \cup A_i) = B \cup \left(\bigcap_{i=1}^{n} A_i\right).$$

28. Довести, що $\{\{x\}, \{x, y\}\} = \{\{z\}, \{z, u\}\} \Leftrightarrow x = z$ і $y = u$.

29. Довести, що

a) $A \times (B \times C) \neq (A \times B) \times C$,

b) $(A \cap B) \times (C \cap D) = (A \times C) \cap (B \times D)$.

30. Надати геометричну інтерпретацію відношень:

a) $(x, y) \in D^2 | y = x\}$, b) $\{(x, y) \in D^2 | y > x\}$,

c) $\{(x, y) \in D^2 | 0 \leq x \leq 1$ або $0 \leq y \leq 1\}$.

31. Знайти R^2 і R^{-1} для відношення $R = \{(x, y) | x, y \in N^+$ і ділить$\}$.

32. Довести, що для довільних бінарних відношень мають місце рівності:

a) $(R \cup R_1)^{-1} = R^{-1} \cup R_1^{-1}$,

b) $(R')^{-1} = (R^{-1})'$.

Довести, що коли бінарне відношення R на A

a) симетричне, то $R = R^{-1}$;

b) рефлексивне і транзитивне, то $R^2 = R$;

c) рефлексивне і антисиметричне, то $R \cap R^{-1} = i$.

33. Навести приклад бінарного відношення, яке

a) рефлексивне, симетричне, нетранзитивне;

b) рефлексивне, несиметричне, транзитивне;

c) рефлексивне, антисиметричне, нетранзитивне;

d) нерефлексивне, симетричне, транзитивне.

34. Нехай множина $A = \{a, b, c, d, e\}$, а відношення

$R = \{(a, b), (a, c), (b, b), (c, c), (e, e), (d, d), (d, a), (d, b), (c, d), (e, d)\}$,

$R_1 = \{(a, d), (d, a), (c, d), (c, c), (b, b)\}$ — бінарні відношення на A.

Побудувати (a) R', (b) $R \cap R_1$, (c) $R \cup R_1$ (d) $R \setminus R_1$, (e) $R \div R_1$.

Чи буде відношення R, $R \cap R_1$, $R \div R_1$, $R \cup R_1$, $R * R_1$

(a) рефлексивним, (b) симетричним, (c) транзитивним?

Побудувати

(а) матрицю відношення R, R_1, $R \cap R_1$, $R \cup R_1$;

(б) відношення R_i,

(в) відношення R_s,

(г) відношення R_t,

(д) матрицю відношення $R_i \cup R_s$,

(е) матриці відношення R, R^2, R^3.

35. Яким відношенням буде транзитивне замикання відношення «X прямий нащадок Y»?

36. Нехай $A = \{a, b\}$ — двохелементна множина. Побудуйте на множині A всі a) бінарні відношення; b) відношення квазіпорядку; c) відношення часткового порядку; d) відношення еквівалентності. Вкажіть серед усіх бінарних відношень на A всі

1) рефлексивні відношення; 2) симетричні відношення;

3) антисиметричні відношення; 4) транзитивні відношення.

37. Нехай π — деяка площина, v — довільна точка на площині π і R — бінарне відношення між точками цієї площини таке, що $aRb \Leftrightarrow va = vb$, де va, vb — відрізки між точками v і a, v і b відповідно. Показати, що R є відношенням еквівалентності.

38. Показати, що відношення $m|n$ (m ділить націло n) на множині натуральних чисел N є відношенням часткового порядку.

39. а) Нехай $A = \{2, 3, 5, 6, 11, 12, 14\}$ і $|$ — відношення часткового порядку на A (див. попередній приклад). Побудувати множину $|$ і упорядкувати множину A відносно цього порядку.

б) Якою буде матриця відношення $i \le j$, яке задане на множині $\{1, 2, 3,..., n\}$.

в) Довести, що в частково упорядкованій скінченній множині існують максимальний і мінімальний елементи.

г) Довести, що в частково упорядкованій множині максимальний елемент буде мінімальним, якщо він не порівнюється ні з одним елементом цієї множини.

40. Показати, що відношення $\subseteq$ для множин є відношенням часткового порядку.

Для яких множин A булеан B (A) буде лінійно упорядкованою множиною щодо відношення $\subseteq$?

41. Показати, що відношення $m|n$ на множині цілих чисел Z є відношенням квазіпорядку.

42. Нехай R — бінарне відношення на множині дійсних чисел D, задане нерівністю

$$aRb \Leftrightarrow \frac{a}{a^2+1} \le \frac{b}{b^2+1}.$$

Показати, що R є відношенням квазіпорядку.

43. Нехай R — бінарне відношення на множині дійсних чисел з інтервалу $(1, \infty)$ таке, що для довільних $a, b \in (1, \infty)$

$$aRb \Leftrightarrow \frac{a}{a^2+1} \le \frac{b}{b^2+1}.$$

Показати, що R є відношенням часткового порядку.

Чи буде це відношення частковим порядком, якщо його розглядати на інтервалі a) $[1,\infty)$; b) $(-1, 1)$; c) $[-1,1]$; d) $(-\infty, 1]$; e) $(-\infty, 0)$.

44. Нехай $<$ — довільне іррефлексивне і транзитивне відношення, визначене на множині A. Нехай $x \le y$ означає, що $x = y$ або $x < y$. Довести, що відношення $\le$ є відношенням часткового порядку.

45. Нехай α — відношення, визначене на множині $N \times N$ таким чином: $(m, n)\ \alpha\ (m', n') \Leftrightarrow m \le m'$ і $n \le n'$ в N. Довести, що

а) відношення α буде відношенням часткового порядку;

б) довільна непуста підмножина множини $N \times N$ має мінімальний елемент.

46. Нехай A і B — дві частково упорядковані множини відношенням $\le$. Визначимо лексикографічний порядок на декартовому добутку $A \times B$ таким чином: $(x, y) \le (x', y') \Leftrightarrow x < x'$ або $x = x'$ у множині A і $y < y'$ у множині B. Довести, що множина $A \times B$ буде частково упорядкованою відносно цього порядку.

47. Довести, що коли f — функція із A в B, а g — функція із B в C, то $f * g$ — функція із A в C.

48. Довести, що для того щоб відображення $R : A \to B$ було взаємно однозначним, необхідно і достатньо, щоб $i_A = R * R^{-1}$, $R^{-1} * R = i_B$.

49. Довести, що коли f — довільна функція, то

$$f(A \cap B) \subseteq f(A) \cap f(B); \quad (A \subseteq B) \Rightarrow f(A) \subseteq f(B).$$

50. Встановити взаємно однозначну відповідність між множинами $A \times B$ і $B \times A$.

51. Довести наслідки 4 і 6.

52. Довести, що множина дійсних чисел незліченна.

53. Довести методом математичної індукції, що

- $n^5 - n$ кратне 5, $n^7 - n$ кратне 7,
- $n \cdot (2n^2 - 3n + 1)$ кратне 6, $5 \cdot 2^{3n-2} + 3^{3n-1}$ кратне 19,
- $1^3 + 2^3 + \ldots + n^3 = (n \cdot (n + 1)/2)^2$,
- $1^2 + 2^2 + \ldots + (2n - 1)^2 = n \cdot (4n^2 - 1)/3$,
- $(1 - 1/4) \cdot (1 - 1/9) \cdot \ldots \cdot (1 - 1/(n + 1)^2) = (n + 2)/(2n + 2)$,
- $1/(n + 1) + 1/(n + 2) + \ldots + 1/2n > (13/24)$,
- $1 \cdot 2 + 2 \cdot 5 + \ldots + n \cdot (3n - 1) = n^2 \cdot (n + 1)$,
- $1/2 \cdot 3/4 \cdot 5/6 \cdot \ldots \cdot (2n - 1)/2n < 1/\sqrt{3n+1}$,
- при $n \ge 2$ число $2^{2^n} + 1$ закінчується числом 7.

54. Довести, що коли $x_1, x_2, \ldots, x_n$ — додатні дійсні числа, такі що $x_1 \cdot x_2 \cdot \ldots \cdot x_n = 1$, то $x_1 + x_2 + \ldots + x_n \ge n$. Користуючись доведеною нерівністю, довести нерівність

$$\frac{x_1 + x_2 + \ldots + x_n}{n} \ge \sqrt[n]{x_1 x_2 \ldots x_n}.$$

55. Довести, що коли $x_1, x_2, \ldots, x_n$ додатні дійсні числа такі, що $x_1 + x_2 + \ldots + x_n \le 1/2$, то $(1 - x_1) \cdot (1 - x_2) \cdot \ldots \cdot (1 - x_n) \ge 1/2$.

56. Довести, що довільну суму грошей, більшу від 7 карбованців, можна розміняти купюрами вартістю 3 і 5 карбованців.

57. Побудувати взаємно однозначну відповідність між множинами натуральних і раціональних чисел.

58. Довести, що множина нескінченна тоді і і тільки тоді, коли вона еквівалентна своїй власній підмножині.

59. Довести зліченність множини

a) многочленів з цілими коефіцієнтами від однієї змінної;

b) алгебраїчних чисел, тобто чисел, які являються коренями многочленів з цілими коефіцієнтами від однієї змінної.

61. Довести зліченність множини всіх скінченних підмножин зліченної множини.

62. Довести еквівалентність множин:

a) (0, 1), [0, 1], (0, 1], [0, 1);

b) $[a, b]$ і $[c, d]$, де $a < b$, $c < d$;

c) $[a, b]$ і D.

1.9. Елементи комбінаторики

На практиці часто трапляються задачі, в яких необхідно підрахувати число всіх можливих способів розміщення деяких предметів скінченної множини або число всіх можливих способів виконання певної дії зі скінченної множини таких дій. Наприклад, скількома різними способами можна розставити дужки у виразі $a + b + c + d + e + f$, якщо операція + асоціативна, а літери a, b, c, d, e, f — деякі дійсні числа? Скількома способами можуть бути розподілені медалі на чемпіонаті світу з футболу серед 24 команд-учасниць фінальної групи? Задачі такого типу називаються *комбінаторними*, а методи їх розв'язку називають *методами комбінаторного аналізу*. Оскільки комбінаторика має справу зі скінченними множинами, то її часто називають *теорією скінченних множин*.

1.10. Основне правило комбінаторики

Розглянемо таку задачу. Нехай з пункту A міста Києва в пункт B можна доїхати трьома видами транспорту — тролейбусом (Т), автобусом (А) і метро (М), а з пункту B в пункт C — лише двома видами транспорту: тролейбусом (Т) і автобусом (А). Скількома способами можна доїхати з пункту A в пункт C міста Києва? Розв'язок цієї за-

дачі зводиться, очевидно, до підрахунку числа елементів у декартовому добутку множин $\{A, T, M\} \times \{A, T\}$. Число таких елементів, як відомо, дорівнює добутку числа елементів першої множини на число елементів другої множини, тобто в нашому випадку це $3 \cdot 2 = 6$. Отже, існує шість способів доїхати із пункту A в пункт C. Виявляється, що за цією простою задачею стоїть правило, яке називається **основним правилом комбінаторики.**

Нехай необхідно виконати послідовно k дій. Якщо першу можна виконати n_1 способами, другу — n_2 способами і так далі до k-ї дії, яку можна виконати n_k способами, то всі k дій можна виконати $n_1 \cdot n_2 \cdot \ldots \cdot n_k$ способами.

Справедливість цього правила безпосередньо випливає з теореми 17.

Приклад. 1. Скількома способами на першості світу з футболу можуть бути розподілені медалі, якщо у фінальній частині грають 24 команди?

Розв'язок. Золоту медаль може одержати кожна із 24 команд, тобто маємо 24 можливості. Срібну медаль може виграти одна з 23 команд, а бронзову — одна з 22 команд. За основним правилом комбінаторики загальне число способів розподілу медалей буде $24 \cdot 23 \cdot 22 = 12\,144$.

2. Скільки тризначних чисел можна скласти з цифр 0, 1, 2, 3, 4, 5, якщо:

а) цифри можуть повторюватися;

б) ні одна з цифр не повторюється двічі;

в) цифри непарні і можуть повторюватися.

Розв'язок. а) Першою цифрою може бути одна з цифр 1, 2, 3, 4, 5, оскільки 0 не може бути першою цифрою, тому що в такому випадку число не буде тризначним. Якщо перша цифра вибрана, то друга може бути вибрана шістьма способами, як і третя цифра. Отже, загальне число тризначних чисел $5 \cdot 6 \cdot 6 = 180$.

б) Першою цифрою може бути одна з п'яти цифр — 1, 2, 3, 4, 5, якщо перша цифра вибрана, то другою може бути теж одна з п'яти цифр (тут уже враховується 0), а третя може бути вибрана 4 способами із чотирьох цифр, що залишилися. Отже, загальна кількість таких тризначних чисел $5 \cdot 5 \cdot 4 = 100$.

в) Першою цифрою може бути одна з трьох цифр 1, 3, 5. Другою теж може бути одна з цих трьох цифр. Аналогічно і третя цифра може бути вибрана трьома способами. Таким чином, загальна кількість таких чисел дорівнює $3 \cdot 3 \cdot 3 = 27$ ♠

1.10.1. Число різних k-елементних підмножин n-елементної множини

З'ясуємо тепер, скільки існує різних підмножин із k елементів у множині, що має n елементів ($k < n$).

Теорема 23. *Число різних k-елементних підмножин n-елементної множини дорівнює*

$$C_n^k = \frac{n!}{k!(n-k)!}, \qquad (1)$$

де скорочення $n! = n \cdot (n-1) \cdot \ldots \cdot 3 \cdot 2 \cdot 1$ *називається* **факторіалом числа** n (*читається n-факторіал*).

Доведення. Щоб побудувати k-елементну підмножину множини A, необхідно до $(k-1)$-елементної множини приєднати один із $n-k+1$ елементів, які не входять у цю підмножину. Оскільки число $(k-1)$-елементних підмножин дорівнює C_n^{k-1} і кожну з цих підмножин можна зробити k-елементною $n-k+1$ способами, то згідно з основним правилом комбінаторики одержуємо число $C_n^{k-1} \cdot (n-k+1)$ підмножин. Але не всі ці підмножини будуть різними, оскільки довільну k-елементну множину можна так побудувати k способами. Отже,

$$C_n^k = \frac{n-k+1}{k} C_n^{k-1} = \frac{(n-k+1)(n-k+2)}{k \cdot (k-1)} C_n^{k-2} = \ldots =$$

$$= \frac{(n-k+1)(n-k+2)\ldots(n-1)}{k \cdot (k-1) \cdot \ldots \cdot 2} C_n^1.$$

Оскільки C_n^1 — число одноелементних підмножин множини A дорівнює n, то

$$C_n^k = \frac{n(n-1)\ldots(n-k+1)}{k!} = \frac{n!}{k!(n-k)!}. \blacksquare$$

Довільна k-елементна підмножина множини A називається **комбінацією, або сполученням,** а число C_n^k — числом комбінацій або сполучень із n елементів по k елементів.

Числа C_n^k називають **біноміальними коефіцієнтами.** Зміст цієї назви буде зрозумілим дещо пізніше.

Біноміальні коефіцієнти мають цікаву геометричну інтерпретацію. Нехай маємо прямокутну шахову дошку розміром m на n, розміщену на координатній площині так, як це показано на рис. 1.9.1.

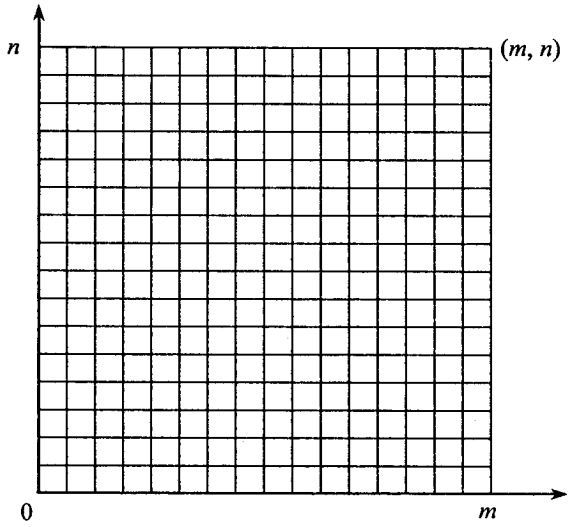


Рис. 1.9.1. Інтерпретація біноміальних коефіцієнтів

Ця дошка складається з $m \cdot n$ елементарних квадратів, поділених $n - 1$ горизонтальними лініями і $m - 1$ вертикальними. Визначимо, скількома різними найкоротшими шляхами можна потрапити з точки (0, 0) у точку (m, n) на цій дошці.

Кожний найкоротший шлях, який веде з точки (0, 0) у точку (m, n), складається, очевидно, з $m + n$ сторін елементарних квадратів, серед яких — m горизонтальних і n вертикальних. Ці шляхи відрізняються між собою лише числом вертикальних і горизонтальних сторін. Отже, загальна кількість шляхів дорівнює числу способів, якими із $m + n$ сторін можна вибрати n вертикальних, тобто це число дорівнює C_{m+n}^{n}.

Зауважимо, що можна було б вести підрахунки не за вертикальними сторонами, а за горизонтальними. Тобто, існує C_{m+n}^{m} різних найкоротших шляхів і, отже, справедлива рівність

$$C_{m+n}^{n} = C_{m+n}^{m} \text{ (}\textit{формула симетрії}\text{).}$$

Наслідок 7. $C_n^k = C_{n-1}^k + C_{n-1}^{k-1}$ (формула додавання).

Доведення.

$$C_{n-1}^k + C_{n-1}^{k-1} = \frac{(n-1)!}{k!(n-k-1)!} + \frac{(n-1)!}{(k-1)!(n-k)!} =$$

$$= (n-1)!\left(\frac{n-k}{k!(n-k)!} + \frac{k}{k!(n-k)!}\right) = \frac{(n-1)!\cdot n}{k!(n-k)!} = \frac{n!}{k!(n-k)!} = C_n^k. \blacksquare$$

Приклад. 1. Збірна команда університету з волейболу налічує 15 осіб. Скільки різних варіантів повинен розглянути тренер перед грою, щоб заявити список гравців на гру?

Розв'язок. Число гравців волейбольної команди дорівнює шести. Отже, число всіх можливих варіантів — це число різних підмножин, які складаються з шести елементів у множині із 15 елементів. Отже, згідно з теоремою 23 маємо

$$C_{15}^{6} = \frac{15!}{6!9!} = \frac{15 \cdot 14 \cdot 13 \cdot 12 \cdot 11 \cdot 10}{6 \cdot 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1} = 5 \cdot 7 \cdot 13 \cdot 11 = 5005.$$

2. У скількох точках перетинаються діагоналі випуклого десятикутника, якщо будь-які три з них не перетинаються в одній точці?

Розв'язок. Кожній точці перетину двох різних діагоналей відповідає чотири вершини десятикутника, а кожним чотирьом вершинам — одна точка перетину. Таким чином, число всіх точок перетину дорівнює числу способів, якими із 10 вершин можна вибрати чотири вершини, тобто

$$C_{10}^{4} = \frac{10!}{4!6!} = \frac{10 \cdot 9 \cdot 8 \cdot 7}{4 \cdot 3 \cdot 2} = 210$$

точок перетину ♠

1.10.2. Число підмножин даної множини

Нехай $A = \{a_1, a_2, \ldots, a_n\}$ — деяка скінченна множина, елементи якої перенумеровані. Під час роботи зі скінченними множинами на обчислювальних машинах такі множини часто задають за допомогою *характеристичних векторів*.

Нехай $A' \subseteq A$ — довільна підмножина множини A. Характеристичний вектор $v(A') = (v_1, v_2, \ldots, v_n)$ для множини A' визначається за допомогою такої відповідності:

$$v_i = \begin{cases} 1, & \text{якщо елемент } a_i \text{ із } A \text{ належить множині } A'; \\ 0, & \text{якщо елемент } a_i \text{ із } A \text{ не належить множині } A'. \end{cases}$$

Наприклад, якщо $A = \{a, b, c, d, e, f\}$, і $A' = \{a, e, f\}$, то $v(A') = (1, 0, 0, 0, 1, 1)$.

Теорема 24. *$A' = A'' \Leftrightarrow v(A') = v(A'')$, де A', A'' — деякі підмножини множини A.*

Доведення. Нехай v' і v'' — характеристичні вектори множин A' і A'' відповідно. Якщо $A' = A''$, то для довільного i маємо $v'_i = v''_i$. Звідси випливає, що $v'(A') = v''(A'')$.

Навпаки, якщо $v'(A'') = v''(A'')$, то для довільного i маємо $v'_i = v''_i$. За визначенням характеристичного вектора одержуємо, що якщо $v'_i = v''_i = 1$, то a_i належить як підмножині A', так і підмножині A'', а якщо $v'_i = v''_i = 0$, то елемент a_i не належить ні тій, ні другій підмножині. Звідси випливає, що підмножини A' і A'' складаються з одних і тих самих елементів, тобто $A' = A''$. ■

Наслідок 8. *Число різних підмножин n-елементної множини дорівнює* 2^n.

Дійсно, оскільки число компонент характеристичного вектора дорівнює n і кожна компонента може приймати одне з двох значень — 0 або 1, — то число різних можливих характеристичних векторів, за основним правилом комбінаторики, дорівнює 2^n.

Наслідок 9. $\sum_{k=0}^{n} C_n^k = 2^n$, *де* $0! = 1$.

Дійсно, оскільки C_n^k — число різних k-елементних підмножин n-елементної множини, то сума всіх таких чисел є числом усіх підмножин даної n-елементної множини.

1.10.3. Перестановки і розміщення упорядкованих множин

Комбінаторні формули, які були встановлені вище, не залежали від порядку елементів у множинах. Формули, які будуть встановлені нижче, відносяться до упорядкованих множин. Дві упорядковані множини вважаються різними, якщо вони відрізняються між собою або своїми елементами, або порядком елементів. Зрозуміло, що коли множина має більше одного елемента, то її можна упорядкувати більше ніж одним способом. З'ясуємо, скількома різними способами можна упорядкувати скінченну n-елементну множину A.

Упорядковані множини, які відрізняються одна від одної лише порядком своїх елементів (тобто можуть бути одержані з однієї й тієї ж множини), називаються **перестановками.** Позначимо число всіх перестановок n-елементної множини P_n.

Теорема 25. $P_n = n!$.

Доведення. Будемо послідовно вибирати елементи множини A і розміщувати їх у заданому порядку на n місцях. На перше місце мож-

на покласти довільний із n елементів множини A, тобто маємо n можливостей. Після того як перше місце заповнилося, на друге місце можна покласти довільний із $n-1$ елементів і т. д. За основним правилом комбінаторики маємо $P_n = n \cdot (n-1) \cdot \ldots \cdot 3 \cdot 2 \cdot 1 = n!$ можливостей упорядкування n-елементної множини A. ■

Приклад. 1. Скількома різними способами можна розмістити п'ять книжок на книжковій полиці?

Розв'язок. Шукане число розміщень є числом способів упорядкування множини із п'яти елементів. Значить, це число дорівнює $P_5 = 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1 = 120$.

2. Скількома способами можна упорядкувати n-елементну множину $A = \{1, 2, \ldots, n\}$ $(n \geq 2)$ так, щоб останні два елементи були $n-1$ і n?

Розв'язок. Число способів буде $P_{n-2} = (n-2)!$ ♠

1.10.4. Перестановки з повтореннями

Розглянемо таку задачу. Скількома способами можна представити n-елементну множину A у вигляді об'єднання її підмножин A_1, $A_2, \ldots, A_m$, що попарно не перетинаються, так, що $|A_1| = k_1$, $|A_2| = k_2$, …, $|A_m| = k_m$, $k_i \geq 0$, $k_1 + k_2 + \ldots + k_m = n$.

Поставлену задачу можна розв'язати таким чином. Візьмемо довільну k_1-елементну підмножину A_1 множини A (це можна зробити $C_n^{k_1}$ способами); серед $n - k_1$ елементів, що залишилися, візьмемо k_2-елементну підмножину A_2 множини A (це можна зробити $C_{n-k_1}^{k_2}$ способами) і т. д. За основним правилом комбінаторики маємо, що число шуканих представлень множини A дорівнює

$$C_n^{k_1} \cdot C_{n-k_1}^{k_2} \ldots C_{n-k_1-k_2-\ldots-k_{m-1}}^{k_m} = \frac{n!}{k_1!(n-k_1)!} \cdot \frac{(n-k_1)!}{k_2!(n-k_1-k_2)!} \times \ldots \times$$

$$\times \frac{(n-k_1-k_2-\ldots-k_{m-1})!}{k_m!(n-k_1-k_2-\ldots-k_{m-1}-k_m)!} = \frac{n!}{k_1!k_2!\ldots k_m!}.$$

Отже, має місце така теорема [12].

Теорема 26. *Нехай $k_1, k_2, \ldots, k_m$ — деякі натуральні числа такі, що $k_1 + k_2 + \ldots + k_m = n$. Кількість способів, якими можна представити n-елементну множину у вигляді об'єднання її підмножин A_1, $A_2, \ldots, A_m$, число елементів яких відповідно $k_1, k_2, \ldots, k_m$, дорівнює*

$$C_n(k_1, k_2, \ldots, k_m) = \frac{n!}{k_1!k_2!\ldots k_m!}. \qquad (2)$$

Числа $C_n(k_1, k_2,..., k_m)$ називаються **поліноміальними коефіцієнтами.**

Розглянемо одну корисну інтерпретацію теореми 26. Нехай маємо деякий алфавіт $X = \{a, b, c,..., d\}$ із m символів і множину із n символів цього алфавіту, причому серед n елементів цієї множини міститься k_1 екземплярів літери a, k_2 — літери b, ..., k_m — літери d і $k_1 + k_2 + ... + k_m = n$. Необхідно визначити кількість різних слів, які можна побудувати із цих n літер.

Перенумеруємо місця числами від 1 до n, на яких стоять літери. Кожне слово однозначно визначається множинами A_1 (номери місць, на яких стоїть літера a), A_2 (номери місць, на яких стоїть літера b),..., A_m (номери місць, на яких стоїть літера d). Отже, число різних слів дорівнює числу різних представлень n-елементної множини $\{1, 2,..., n\}$ у вигляді об'єднання її підмножин $A_1, A_2,..., A_m$, тобто

$$C_n(k_1, k_2, ..., k_m) = \frac{n!}{k_1!k_2!...k_m!}.$$

З цієї задачі випливає інша формулювання попередньої теореми.

Теорема 27. *Число різних перестановок, які можна побудувати на n елементах, серед яких знаходиться k_1 елементів першого типу, k_2 елементів другого типу, ..., k_m елементів m-го типу, дорівнює $C_n(k_1, k_2,..., k_m)$.*

Приклад. 1. Скільки різних слів можна побудувати перестановкою літер у слові «лаваш»?

Розв'язок. Слово «лаваш» включає по одному екземпляру літер л, в, ш і два екземпляри літери а, загальна ж кількість літер дорівнює 5. За встановленою вище формулою знаходимо

$$\frac{5!}{2!1!1!1!} = 5 \cdot 4 \cdot 3 = 60.$$

2. Скільки слів довжини 8 можна скласти з літер a і b таких, що кількість літер a в цих словах не перевищує три?

Розв'язок. Такими словами будуть усі слова, які не мають ні однієї літери a, усі слова, які мають лише одну літеру a, усі слова, які мають дві літери a і, нарешті, усі слова, які мають три літери a, тобто загальна кількість слів дорівнює

$$C_8(0, 8) + C_8(1, 7) + C_8(2, 6) + C_8(3, 5) =$$
$$= \frac{8!}{0!8!} + \frac{8!}{1!7!} + \frac{8!}{2!6!} + \frac{8!}{3!5!} = 1 + 8 + 28 + 56 = 93. \blacktriangle$$

1.10.5. Розміщення елементів множини

Нехай дано деяку неупорядковану n-елементну множину A. Скільки різних упорядкованих k-елементних підмножин може мати множина A? Розглянемо два можливі варіанти цієї задачі:

а) підмножина має k різних між собою елементів;

б) підмножина має k не обов'язково різних між собою елементів.

Отже, у задачі а) підмножини задаються ненадлишково, а в задачі б) — надлишково, але число всіх різних елементів підмножини разом з числом усіх екземплярів кожного з її елементів дорівнює k.

Упорядкована k-елементна підмножина множини A, усі елементи якої різні між собою, називається **розміщенням без повторень,** а довільна упорядкована k-елементна підмножина множини A, усі k елементів якої не обов'язково різні між собою, називається **розміщенням з повтореннями.** Зауважимо, що в першому випадку $k \le n$, причому якщо $k = n$, то в цьому випадку розміщення є перестановкою. У другому випадку k не обов'язково має бути менше n, тобто можливо, що $k \ge n$.

Розглянемо першу задачу. Оскільки множина A неупорядкована, то довільна її k-елементна підмножина може бути упорядкована одним із $k!$ способів, а число всіх можливих різних k-елементних підмножин множини A дорівнює C_n^k. Отже, число всіх можливих розміщень із n елементів по k дорівнює $k! \cdot C_n^k$, тобто має місце така теорема.

Теорема 28. *Кількість упорядкованих k-елементних підмножин n-елементної множини, усі k елементів якої різні, дорівнює*

$$A_n^k = k! \cdot C_n^k = \frac{n!}{(n-k)!} = n \cdot (n-1) \cdot \ldots \cdot (n-k+1).$$

Приклад. 1. Студенту необхідно скласти три екзамени протягом семи днів. Скількома способами це можна зробити?

Розв'язок. Шукане число способів дорівнює кількості 3-елементних упорядкованих підмножин множини із 7 елементів, тобто існує $A_7^3 = 7 \cdot 6 \cdot 5 = 210$ способів.

Якщо відомо, що останній екзамен буде складатися на сьомий день, то число способів дорівнює $3 \cdot A_6^2 = 3 \cdot 6 \cdot 5 = 90$.

2. Скількома різними способами можна розмістити 5 студентів у аудиторії, яка має 20 місць?

Розв'язок. Шукане число способів дорівнює числу розміщень із 20 елементів по 5 елементів, тобто $A_{20}^5 = 20 \cdot 19 \cdot 18 \cdot 17 \cdot 16 = 1\,860\,480$. ♠

Розглянемо другу задачу. Нехай $B = \{b_1, b_2, \ldots, b_k\}$ — деяка скінченна k-елементна множина, а $A = \{a_1, a_2, \ldots, a_n\}$ — n-елементна множина і $f: B \to A$ — функція із B в A. Як відомо, функцію f можна задати за допомогою таблиці значень

b_1	b_2	...	b_k
a_{i_1}	a_{i_2}	...	a_{i_k}

де $a_{i_j} = f(b_j)$, $j = 1, 2, \ldots, k$. Тепер задачу можна сформулювати так: скільки існує функцій із множини B в множину A? У такому формулюванні задача розв'язується досить просто.

Умовимося називати кортежом довжини k елементи вигляду $(a_1, a_2, \ldots, a_k)$, де a_i — не обов'язково різні елементи деякої скінченної множини A. Оскільки кожний елемент a_{i_j} може бути довільним елементом множини A, то число різних кортежів вигляду $(a_{i_1}, a_{i_2}, \ldots, a_{i_k})$ може бути n^k.

Теорема 29. *Кількість різних упорядкованих k-елементних підмножин n-елементної множини, усі k елементів якої не обов'язково різні між собою, дорівнює n^k.*

Приклад. Скільки різних слів можна скласти
a) в алфавіті $X = \{0, 1\}$ з вісьми літер;
b) в алфавіті $X = \{0, 1\}$ з 16 літер?

Розв'язок. a) Усіх таких слів буде стільки, скільки існує відображень восьмиелементної множини в множину із двох елементів, тобто за теоремою 29, — $2^8 = 256$ слів.
b) У цьому алфавіті маємо $2^{16} = 65\,536$ слів. ♠

1.10.6. Комбінації елементів з повтореннями

Нехай маємо неупорядковану n-елементну множину A, елементи якої розбиті на n класів (у кожному класі знаходиться по одному елементу), які будуть називатися *типами елементів*.

Визначення 30. **Комбінацією** *або* **сполученням** *із n елементів по m елементів з повтореннями називається m-елементна підмножина множини A, кожний елемент якої належить одному із n типів. Сукупність таких підмножин називають* **комбінаціями** *або* **сполученнями** *із n елементів по m.*

Теорема 30. *Кількість різних комбінацій із n елементів по m елементів з повтореннями дорівнює* $C_{n+m-1}^{n-1} = C_{n+m-1}^{m}$.

Доведення. «Закодуємо» кожну комбінацію таким чином: якщо комбінація включає k_1 елементів першого типу, то записуємо підряд k_1 одиниць, ставимо нуль і після нього ставимо підряд k_2 одиниць, якщо комбінація включає k_2 елементів другого типу і т. д. Наприклад, якщо $A = \{a, b, c, d\}$, то комбінаціям по два елементи з повтореннями відповідають пари $\{a, a\}$, $\{a, b\}$, $\{a, c\}$, $\{a, d\}$, $\{b, b\}$, $\{b, c\}$, $\{c, d\}$, $\{c, c\}$, $\{c, d\}$, $\{d, d\}$, а їх «кодами» будуть відповідно

$$11000,\ 10100,\ 10010,\ 10001,...,\ 00011.$$

Неважко переконатися, що між «кодами» і комбінаціями існує взаємно однозначна відповідність (переконайтеся). Отже, кожній комбінації із n елементів по m відповідає послідовність із m одиниць і $n - 1$ нулів (у розглянутому прикладі $n = 4$, $m = 2$). Отож, число всіх комбінацій із n елементів по m з повтореннями дорівнює числу послідовностей, що складаються із m одиниць і $n - 1$ нулів, тобто дорівнює $C_{n+m-1}^{n-1} = C_{n+m-1}^{m}$. ■

Приклад. Скільки цілих невід'ємних розв'язків має рівняння

$$x_1 + x_2 + ... + x_n = m? \qquad (3)$$

Розв'язок. Розв'язки даного рівняння можна інтерпретувати так. Якщо маємо цілі невід'ємні числа $x_1 + x_2 + \ldots + x_n$, такі що $x_1 + x_2 + \ldots + x_n = m$, то можна скласти комбінацію із n елементів по m, взявши x_1 елементів першого типу, x_2 елементів другого типу, ... , x_n — елементів n-го типу.

Навпаки, маючи комбінацію із n по m елементів, одержимо розв'язок рівняння (3) (x_1 — число елементів першого типу, x_2 — число елементів другого типу,..., x_n — число елементів n-го типу), де всі x_1 невід'ємні ($i = 1, 2,..., n$). Таким чином, між множиною всіх невід'ємних розв'язків рівняння (3) і множиною всіх комбінацій із n елементів по m встановлюється взаємно однозначна відповідність. Отже, число цілих невід'ємних розв'язків рівняння (3) дорівнює $C_{n+m-1}^{n-1} = C_{n+m-1}^{m}$.

Наприклад, якщо $x_1 + x_2 + x_3 + x_4 = 10$, то це рівняння має

$$C_{10+4-1}^{4} = C_{13}^{4} = \frac{13!}{4!9!} = 715$$

цілих невід'ємних розв'язків. ♠

1.11. Біном Ньютона

З елементарної математики добре відомі формули скороченого множення:

$$(a+b)^2 = a^2 + 2ab + b^2 \text{ і } (a+b)^3 = a^3 + 3a^2b + 3ab^2 + b^3.$$

Ці формули можна записати і так:

$$(a+b)^2 = C_2^0 a^2 b^0 + C_2^1 ab + C_2^2 a^0 b^2,$$
$$(a+b)^3 = C_3^0 a^3 b^0 + C_3^1 a^2 b + C_3^2 ab^2 + C_3^3 a^0 b^3.$$

Виявляється, що має місце і загальна закономірність.

Теорема 31. *Справедлива рівність*

$$(a+b)^n = C_n^0 a^n b^0 + C_n^1 a^{n-1} b^1 + C_n^2 a^{n-2} b^2 + \ldots + C_n^n a^0 b^n. \qquad (4)$$

Доведення проводиться індукцією за числом n.

$n = 1$. $(a+b)^1 = C_1^0 a^1 b^0 + C_1^0 a^0 b^1 = a + b$. Отже, база індукції має місце.

Нехай теорема справедлива для $m = n$. Покажемо, що вона справедлива і для $m = n + 1$. За припущенням індукції маємо

$$(a+b)^{n+1} = (a+b)^n (a+b) = (C_n^0 a^n b^0 + C_n^1 a^{n-1} b + \ldots C_n^n a^0 b^n)(a+b) =$$
$$= C_n^0 a^{n+1} b^0 + (C_n^1 a^n b + C_n^0 a^n b) + (C_n^2 a^{n-1} b^2 + C_n^1 a^{n-1} b^2) + \ldots + C_n^n a^0 b^{n+1} =$$
$$= C_{n+1}^0 a^{n+1} b^0 + C_{n+1}^1 a^n b + \ldots + C_{n+1}^n a^1 b^n + C_{n+1}^{n+1} a^0 b^{n+1},$$

згідно з наслідком 7 і тим, що $C_n^0 = C_{n+1}^0 = 1$. ■

Рівність (4) називають **біномом Ньютона.** З рівності, встановленої в наслідку 7, випливає, що біноміальні коефіцієнти (тепер зрозуміло, чому ми їх назвали так раніше) можна виписати у вигляді трикутної таблиці, яка носить назву *трикутника Паскаля.*

1 1 $n = 1$
1 2 1 $n = 2$
1 3 3 1 $n = 3$
1 4 6 4 1 $n = 4$
1 5 10 10 5 1 $n = 5$
.....

В n-му рядку трикутника Паскаля стоять коефіцієнти розкладу (4), причому кожний коефіцієнт, крім двох крайніх, які дорівню-

ють 1, — це сума відповідних коефіцієнтів із попереднього рядка. З властивостями цього трикутника, а також з іншими корисними властивостями біноміальних коефіцієнтів можна ознайомитися в [5, 6, 12, 32, 33, 40, 43].

1.11.1. Поліноміальна теорема

Наступна теорема є узагальненням біному Ньютона і дає відповідь, як розкривати дужки при обчисленні виразу вигляду

$$(a_1 + a_2 + ... + a_k)^n.$$

Теорема 32 (*Поліноміальна теорема*). *Має місце така рівність*:

$$(a_1 + a_2 + ... + a_k)^n = \sum_{r_1 \geq 0,...,r_k \geq 0 \ \& \ r_1+...+r_k=n} C_n(r_1, r_2..., r_k)\, a^{r_1} a^{r_2} ... a^{r_k}.$$

Доведення. Перемножимо послідовно $a_1 + a_2 + ... + a_k$ n разів. У результаті одержимо k^n доданків вигляду $d_1 d_2 \ ... \ d_n$, де кожний множник d_i дорівнює або a_1, або a_2,..., або a_k. Нехай $B(r_1, r_2,..., r_k)$ означає множину всіх таких доданків, у яких a_1 трапляється r_1 разів, a_2 – r_2 разів, ..., a_k – r_k разів. Як відомо (див. теорему 26), число таких доданків дорівнює числу способів розбиття множини $\{1, 2,..., n\}$ на k підмножин $B_1, B_2,..., B_k$ так, щоб у множині B_s було r_s елементів ($r_s \geq 0$), тобто дорівнює $C_n(r_1, r_2,..., r_k)$, де $r_1 + r_2 + ... + r_k = n$, а B_s — це множина тих значень i, для яких $d_i = a_s$. Отже,

$$(a_1 + a_2 + ... + a_k)^n = \sum_{r_1 \geq 0,...,r_k \geq 0 \ \& \ r_1+...+r_k=n} C_n(r_1, r_2..., r_k)\, a^{r_1} a^{r_2} ... a^{r_k}. \ \blacksquare$$

З поліноміальної теореми, як окремий випадок, випливає виведена вище формула біному Ньютона.

Наслідок 10. $(a + b)^n = \sum_{k=0}^{n} C_n^k a^{n-k} b^k$.

Дійсно, якщо припустити, що $r_1 = k$, то $r_2 = n - k$ і з поліноміальної теореми випливає даний наслідок. ■

1.11.2. Властивості біноміальних коефіцієнтів

Раніше були встановлені деякі властивості біноміальних коефіцієнтів. Нагадаємо їх:

- а) формула симетрії $C_{m+n}^{n} = C_{m+n}^{m}$,

- b) формула додавання $C_n^k = C_{n-1}^k + C_{n-1}^{k-1}$,
- c) формула суми всіх біноміальних коефіцієнтів $C_n^0 + C_n^1 + ... + C_n^n = 2^n$.

Користуючись цими властивостями, неважко одержати іще ряд формул. Наприклад, якщо покласти в біномі Ньютона $a = 1$, $b = -1$, то одержимо таку властивість

$$\text{d) } C_n^0 - C_n^1 + C_n^2 - ... + (-1)^n C_n^n = 0.$$

Наступну формулу легко можна встановити, виконуючи обчислення:

$$\text{e) } C_n^k = \frac{n}{k} \cdot C_{n-1}^{k-1}.$$

Наведемо ще деякі корисні властивості біноміальних коефіцієнтів:

- f) $C_n^0 + C_{n+1}^1 + ... + C_{n+k}^n = C_{n+k+1}^n$;
- g) $C_{n-1}^{r-1} + C_{n-2}^{r-1} + ... + C_{n-r}^{r-1} = C_n^r$;
- h) $C_n^1 + 2C_n^2 + ... + nC_n^n = n2^{n-1}$;
- i) $C_n^0 C_m^k + C_n^1 C_m^{k-1} + ... + C_n^k C_m^0 = C_{n+m}^k$;
- k) $C_n^m C_k^0 + C_{n-1}^{m-1} C_{k+1}^1 + ... + C_{n-m}^0 C_{k+m}^m = C_{n+k+1}^m$;
- l) $(C_n^0)^2 + (C_n^1)^2 + ... + (C_n^n)^2 = C_{2n}^n$.

На закінчення даного розділу зазначимо, що переліченими властивостями біноміальних коефіцієнтів їх властивості далеко не вичерпуються. Існує велика кількість властивостей цих коефіцієнтів і з'являються все нові й нові властивості. Як зазначає Д. Кнут у своїй книзі *«Искусство программирования для ЭВМ, т.* 1, *С.* 85», за останні роки таких властивостей з'явилося стільки, що вони радують лише тих людей, які їх встановили. Тому дати вичерпну відповідь про властивості біноміальних коефіцієнтів немає ніякої можливості.

Перейдемо до розгляду основних методів комбінаторного аналізу.

1.12. Основні методи комбінаторного аналізу

1.12.1. Метод рекурентних співвідношень

Цей метод, як відомо, дозволяє знаходити значення деякої функції для величини аргументу через менші значення аргументів. Стосовно комбінаторики цей метод дає можливість знаходити розв'язок комбінаторної задачі для n предметів через розв'язок аналогічної за-

дачі з меншим числом предметів за допомогою деякого співвідношення, яке називається **рекурентним співвідношенням.** Проілюструємо цей метод на прикладі підрахунку числа **комбінацій з повтореннями.**

Комбінації з повтореннями. Нехай число комбінацій із n елементів по k елементів з повтореннями дорівнює f_n^k. Кожна комбінація із n по k елементів включає або ні елемент a_n. Число комбінацій, які не включають елемент a_n, дорівнює f_{n-1}^k. Кожна комбінація може бути одержана приєднанням до елемента a_n деякої комбінації із n елементів по $k-1$ елементів, тобто існує f_n^{k-1} таких комбінацій. Отже, усіх комбінацій буде

$$f_n^k = f_n^{k-1} + f_{n-1}^k.$$

Побудоване рекурентне співвідношення дає можливість знайти f_n^k. Дійсно,

$$f_n^k = f_n^{k-1} + \left(f_{n-1}^{k-1} + f_{n-2}^k\right) = f_n^{k-1} + f_{n-1}^{k-1} + \ldots + f_2^{k-1} + f_1^k. \qquad (5)$$

Очевидно, що $f_n^1 = n$ і $f_1^k = 1$.

Припустивши у формулі (5), що $k = 2$, одержимо

$$f_n^2 = f_n^1 + f_n^2 = f_n^1 + f_{n-1}^1 + \ldots + f_2^1 + f_1^1 =$$

$$= n + (n-1) + \ldots + 2 + 1 = \frac{n(n+1)}{2} = C_{n+1}^2.$$

Припустимо тепер у формулі (5), що $k = 3$. Одержимо

$$f_n^3 = C_{n+1}^2 + C_n^2 + \ldots + C_3^2 + C_2^2 = C_{n+2}^3,$$

якщо скористатися формулою (g). Повторюючи ці обчислення k разів, одержуємо остаточну формулу

$$f_n^k = C_{n+k-1}^k.$$

1.12.2. Метод включень і вилучень

Нехай A і B — скінченні множини. З'ясуємо, чому дорівнює число $|A \cup B|$, якщо відомі числа $|A|$ і $|B|$. Основна формула, якою можна скористатися для визначення числа $|A \cup B|$, випливає безпосередньо з визначення операції об'єднання множин:

$$|A \cup B| = |A| + |B| - |A \cap B|. \qquad (6)$$

Дійсно, |A| + |B| є числом, яке одержується при підрахунку спочатку всіх елементів множини A, а потім — усіх елементів множини B. Але при цьому спільні елементи множини A і множини B (а їх буде $|A \cap B|$) рахуються двічі, тобто

$$|A| + |B| = |A \cup B| + |A \cap B|,$$

звідки і випливає рівність (6).

Користуючись формулою (6) і законами алгебри множин, можна одержати формулу для числа елементів довільної скінченної сукупності скінченних множин. Наприклад,

$$|A \cup B \cup C| = |A \cup (B \cup C)| = |A| + |B \cup C| - |A \cap (B \cup C)| =$$
$$= |A| + |B| + |C| - |B \cap C| - |(A \cap B) \cup (A \cap C)| =$$
$$= |A| + |B| + |C| - |B \cap C| - (|A \cap B| + |A \cap C| - |(A \cap B) \cap (A \cap C)| =$$
$$= |A| + |B| + |C| - |A \cap B| - |A \cap C| - |B \cap C| + |A \cap B \cap C|.$$

Має місце загальна теорема.

Теорема 33. *Якщо $A_1, A_2, \ldots, A_n$ — деякі скінченні множини, то*

$$|A_1 \cup A_2 \cup \ldots \cup A_n| = \sum_{1 \le i \le n} |A_n| - \sum_{1 \le i < j \le n} |A_i \cap A_j| + \sum_{1 \le i < j < k \le n} |A_i \cap A_j \cap A_k| -$$
$$\ldots + (-1)^{n-1} \sum_{1 \le i < j < k < \ldots < l \le n} |A_i \cap A_j \cap \ldots \cap A_l|. \qquad (7)$$

Доведення. Щоб довести теорему, необхідно показати, що кожний елемент із множини $A_1 \cup A_2 \cup \ldots \cup A_n$ враховується в правій частині рівності (7) лише один раз. Нехай $a \in A_1 \cup A_2 \cup \ldots \cup A_n$ такий елемент, який входить до складу m множин A_i ($i = 1, 2, \ldots, n$). Тоді елемент a підраховується в правій частині (7)

$$C_m^1 - C_m^2 + C_m^3 - \ldots + (-1)^{m-1} C_n^m$$

разів. Але

$$C_m^1 - C_m^2 + C_m^3 - \ldots + (-1)^{m-1} C_m^m =$$
$$= 1 - \left(1 - C_m^1 + C_m^2 - C_m^3 + \ldots + (-1)^{m-1} C_m^m\right) = 1 - (1-1)^m = 1.$$

А це означає, що кожний елемент a із $A_1 \cup A_2 \cup \ldots \cup A_n$ враховується в правій частині рівності (7) лише один раз. ■

Метод підрахунку за формулою (7), який полягає в послідовному виконанні операцій додавання і віднімання, що чергуються між собою, називається **методом включень і вилучень.**

Приклад. 1. Розглянемо всі перестановки множини чисел $\{1, 2,..., n\}$. Необхідно підрахувати кількість усіх перестановок, в яких хоча б одне число стоїть на своєму місці.

Розв'язок. Нехай A_i означає сукупність тих перестановок, в яких число i стоїть на i-му місці. Тоді $S = |A_1 \cup A_2 \cup ... \cup A_n|$ є шуканим числом. Множину $A_i \cap A_j \cap ... \cap A_k$ складають ті перестановки, в яких на місцях $i, j,..., k$ стоять відповідно числа $i, j,..., k$, а на інших $n-k$ місцях — решта $n-k$ чисел, які упорядковані довільним чином. Отже, $|A_i \cap A_j \cap ... \cap A_k| = (n-k)!$, а

$$\sum_{1\le i<j<...<k\le n} |A_i \cap A_j \cap ... \cap A_k| = C_n^k (n-k)! = \frac{n!}{k!}.$$

З рівності (7) випливає, що

$$S = |A_1 \cup A_2 \cup ... \cup A_n| = n!\left(\frac{1}{1!} - \frac{1}{2!} + \frac{1}{3!} - ... + (-1)^{n-1}\frac{1}{n!}\right).$$

2. Нехай $a_1, a_2, \ldots, a_n$ — взаємно прості натуральні числа, а p — теж деяке натуральне число. Довести, що кількість натуральних чисел, які не перевищують p і не діляться ні на одне з чисел $a_1, a_2, \ldots, a_n$, дорівнює $p - |A_1 \cup A_2 \cup ... \cup A_n|$, де $A_i = \{q \in N \mid q < p \text{ і } q \text{ ділиться на } a_i\}$.

Розв'язок. Очевидно, що $|A_i| = \left[\frac{p}{a_i}\right]$, де $[x]$ — найбільше ціле число, яке не перевищує x. Множина $A_i \cup A_j \cup ... \cup A_k$ — це множина таких чисел $q \le p$, які діляться на $a_i, a_j,..., a_k$. Оскільки числа $a_i, a_j,..., a_k$ взаємно прості, то

$$|A_i \cap A_j \cap ... \cap A_k| = \left|\frac{p}{a_i a_j ... a_k}\right|.$$

З рівності (7) маємо, що кількість усіх чисел, які не перевищують p і діляться хоча б на одне з чисел $a_1, a_2,..., a_n$, дорівнює

$$|A_1 \cup A_2 \cup ... \cup A_n| = \sum_{1\le i\le n}\left|\frac{p}{a_i}\right| - \sum_{1\le i<j\le n}\left|\frac{p}{a_i a_j}\right| + \sum_{1\le i<j<k\le n}\left|\frac{p}{a_i a_j a_k}\right| - ... +$$
$$+ (-1)^{n-1}\left|\frac{p}{a_1 a_2 ... a_n}\right|.$$

Отже, кількість чисел, які не перевищують p і не діляться ні на одне з чисел $a_1, a_2,..., a_n$, дорівнює

$$p-|A_1\cup A_2\cup\ldots\cup A_n|=p-\sum_{1\le i\le n}\left|\frac{p}{a_i}\right|+\sum_{1\le i<j\le n}\left|\frac{p}{a_ia_j}\right|-\sum_{1\le i<j<k\le n}\left|\frac{p}{a_ia_ja_k}\right|+\ldots+$$

$$+(-1)^{n-1}\left|\frac{p}{a_1a_2\ldots a_n}\right|.$$

Що і потрібно було встановити. ♠

Теорема 34. *Нехай* $|A_1\cup A_2\cup\ldots\cup A_n|_m$ *— означає число елементів, які входять до складу рівно* m *множин із* $A_1, A_2,\ldots, A_n$. *Тоді*

$$|A_1\cup A_2\cup\ldots\cup A_n|_m=C_m^m\sum_{1\le i_1<i_2<\ldots<i_m\le n}|A_{i_1}\cap A_{i_2}\cap\ldots\cap A_{i_m}|-$$

$$-C_{m+1}^m\sum_{1\le i_1<i_2<\ldots<i_{m+1}\le n}|A_{i_1}\cap A_{i_2}\cap\ldots\cap A_{i_{m+1}}|+$$

$$+\ldots+(-1)^{n-m}C_n^m\sum_{1\le i_1<i_2<i_3<\ldots<i_n\le n}|A_{i_1}\cap A_{i_2}\cap\ldots\cap A_{i_n}|. \qquad (8)$$

Доведення. Нехай a — довільний елемент, який входить до складу k множин із заданої сукупності множин. Для доведення теореми, як і в попередньому випадку, достатньо показати, що елемент a враховується в правій частині рівності (8) лише один раз, якщо $k=m$, і ні разу, якщо $k\ne m$.

Насамперед зауважимо, що коли $k<m$, то елемент a не враховується в (8) жодного разу, а якщо $k=m$, то він враховується лише раз, оскільки a входить лише в одну із множин вигляду $A_{i_1}\cap A_{i_2}\cap\ldots\cap A_{i_k}$.

Розглянемо випадок, коли $k>m$. При підрахунках за формулою (8) елемент a враховується C_k^m разів у першій сумі

$$\sum_{1\le i_1<i_2<i_3<\ldots<i_m\le k}|A_{i_1}\cap A_{i_2}\cap\ldots\cap A_{i_m}|.$$

C_k^{m+1} разів — у другій сумі

$$\sum_{1\le i_1<i_2<i_3<\ldots<i_{m+1}\le k}|A_{i_1}\cap A_{i_2}\cap\ldots\cap A_{i_{m+1}}|$$

і т. д. C_k^k разів — у сумі

$$\sum_{1\le i_1<i_2<i_3<\ldots<i_k\le k}|A_{i_1}\cap A_{i_2}\cap\ldots\cap A_{i_k}|.$$

Решта сум не враховують елемент a, оскільки він входить лише до k множин. Отже, елемент a враховується у (8)

$$C_m^mC_k^m-C_{m+1}^mC_k^{m+1}+C_{m+2}^mC_k^{m+2}-\ldots+(-1)^{k-m}C_k^mC_k^k$$

разів. На підставі того, що $C_r^m C_n^r = C_n^m C_{n-m}^{n-r}$, знаходимо

$$C_m^m C_k^m - C_{m+1}^m C_k^{m+1} + C_{m+2}^m C_k^{m+2} - \ldots + (-1)^{k-m} C_k^m C_k^k =$$
$$= C_k^m \left(C_{k-m}^{k-m} - C_{k-m}^{k-m-1} + \ldots + (-1)^{k-m} C_{k-m}^0 \right) = C_k^m (1-1)^{k-m} = 0. \blacksquare$$

1.12.3. Метод продуктивних функцій

Це один із найефективніших методів комбінаторного аналізу, оскільки узагальнює інші його методи. Введемо поняття продуктивної функції.

Нехай маємо деяку числову послідовність $a_0, a_1, a_2, \ldots, a_n, \ldots$. Формальна сума вигляду

$$A(s) = a_0 + a_1 s + a_2 s^2 + \ldots + a_n s^n + \ldots = \sum_{n=0}^{\infty} a_n s_n \qquad (9)$$

називається **продуктивною функцією послідовності** $\{a_n\}$.

Ідея методу продуктивних функцій полягає в тому, що коли необхідно знайти всі члени деякої послідовності $\{a_n\}$, то за допомогою рекурентних співвідношень для членів послідовності $\{a_n\}$, або з деяких інших міркувань, обчислюють продуктивну функцію $A(s)$. Після цього, розкладаючи її в ряд і знаходячи коефіцієнти при s^n, знаходять a_n.

Як відомо з математичного аналізу, ряди вигляду (9) можна додавати, віднімати і множити, виконуючи відповідні дії над коефіцієнтами при однакових степенях s. Отже, можна говорити і про суму, різницю і добуток продуктивних функцій.

Приклад. 1. Нехай $a_n = c^n$, де c — деяке дійсне число. Тоді

$$A(s) = \sum_{n=0}^{\infty} c^n s^n = \frac{1}{1-cs}$$

(це буде сума нескінченно спадної геометричної прогресії, якщо $|cs| < 1$).

2. Нехай $a_n = C_m^n$ $(n = 0, 1, 2, \ldots, m)$, тоді

$$A(s) = \sum_{n=0}^{m} C_m^n s^n = (1+s)^m.$$

3. Нехай $a_n = f_m^n$ — число сполучень із m елементів по n. Раніше було виведено, що

$$f_m^n = f_m^{n-1} + f_{m-1}^n, \qquad (10)$$

причому $f_m^0 = 1$. Отже, маємо $A(s) = \sum_{n=0}^{\infty} f_m^n s^n$.

Домножимо (10) на s^n і додамо почленно знайдені рівності. Одержимо

$$\sum_{n=1}^{\infty} f_m^n s^n = s\sum_{n=1}^{\infty} f_m^{n-1} s^{n-1} + \sum_{n=1}^{\infty} f_{m-1}^n s^n = s(A_n(s)-1) + A_{n-1}(s) - 1.$$

Тобто $A_n(s) - 1 = s\,(A_n(s) - 1) + A_{n-1}(s) - 1$, або $A_n(s) = \dfrac{1}{1-s} A_{n-1}(s)$.

Звідси знаходимо, що

$$A_n(s) = \frac{1}{(1-s)^2} A_{n-2}(s) = \ldots = \frac{A_1(s)}{(1-s)^{n-1}}.$$

Знайдемо тепер $A_1(s)$: $A_1(s) = \sum\limits_{n=0}^{\infty} f_1^n s^n = \sum\limits_{n=0}^{\infty} s^n = \dfrac{1}{1-s}$.

Отже, остаточно маємо $A_n(s) = \dfrac{1}{(1-s)^n}$.

Розкладаючи в ряд одержану функцію за допомогою формули бінома Ньютона (як відомо з математичного аналізу, формула бінома Ньютона має місце як для дробових, так і для від'ємних показників степені) знаходимо, що $f_m^n = C_{m-1+n}^n$. ♠

Згорткою двох послідовностей $\{a_n\}$ і $\{b_n\}$ називається послідовність $\{c_n\}$, загальний член якої має такий вигляд

$$c_n = a_0 b_n + a_1 b_{n-1} + \ldots + a_k b_{n-k} + \ldots + a_n b_0.$$

Має місце таке просте твердження.

Теорема 35. *Продуктивна функція згортки двох послідовностей дорівнює добутку продуктивних функцій цих послідовностей.*

Доведення. Нехай $\{c_n\}$ — згортка послідовностей $\{a_n\}$ і $\{b_n\}$, а $A(s) = \sum\limits_{n=0}^{\infty} a_n s^n$, $B(s) = \sum\limits_{n=0}^{\infty} b_n s^n$, $C(s) = \sum\limits_{n=0}^{\infty} c_n s^n$ — продуктивні функції послідовностей $\{a_n\}$, $\{b_n\}$, $\{c_n\}$ відповідно. Необхідно довести, що $C(s) = A(s)\,B(s)$.

Перемножаючи два ряди: $A(s)$ і $B(s)$, знаходимо, що коефіцієнт при s^n в одержаному добутку дорівнює

$$a_0 b_n + a_1 b_{n-1} + \ldots + a_k b_{n-k} + \ldots + a_n b_0,$$

тобто дорівнює c_n. Отже, $A(s)\,B(s) = C(s)$. ■

Приклад. Нехай $a_n = a_{n-1} + a_{n-2}a_1 + \ldots + a_1a_{n-2} + a_{n-1}$ і $A(s) = \sum\limits_{n=0}^{\infty} a_n s^n$ — продуктивна функція для послідовності $\{a_n\}$, про яку відомо, що $A(0) = 1$. Покладемо $a_0 = 1$, тоді можемо записати

$$a_n = a_0a_{n-1} + a_{n-2}a_1 + \ldots + a_1a_{n-2} + a_{n-1}a_0,$$

де $n \geq 1$, і

$$\sum_{n=1}^{\infty} a_n s^n = \sum_{n=1}^{\infty} (a_0a_{n-1} + a_{n-2}a_1 + \ldots + a_1a_{n-2} + a_{n-1}a_0)s^n =$$

$$= s\,(\sum_{n=1}^{\infty} (a_0a_{n-1} + a_{n-2}a_1 + \ldots + a_1a_{n-2} + a_{n-1}a_0)s^{n-1} = s\,(A(s))^2.$$

Оскільки $\sum\limits_{n=1}^{\infty} a_n s^n = A(s) - 1$, то, застосовуючи теорему про згортку двох послідовностей, маємо таке рівняння $A(s) - 1 = s(A(s))^2$.

Розв'язуючи його, знаходимо такі корені

$$A(s) = \frac{1 \pm \sqrt{1-4s}}{2s}.$$

Тоді $A(0) = \lim\limits_{s\to 0} \frac{1-\sqrt{1-4s}}{2s} = 1$, а $\lim\limits_{s\to 0} \frac{1+\sqrt{1-4s}}{2s} = \infty$. Отже, оскільки $A(0) = 1$, то

$$A(s) = \frac{1-\sqrt{1-4s}}{2s}. \qquad (11)$$

Знайдемо тепер коефіцієнт при s^n у розкладі функції (11). Скористаємося для цього формулою бінома Ньютона для функції

$$(1-4s)^{1/2} = 1 + \sum_{k=1}^{\infty} \frac{1/2(1/2-1)\ldots(1/2-k+1)}{k!}(-1)^k\,4^k\,s^k =$$

$$= 1 - \sum_{k=1}^{\infty} C_{2k}^k \frac{1}{2k-1} s^k.$$

Звідси одержуємо, що

$$A(s) = \frac{1-\sqrt{1-4s}}{2s} = 1/2 \sum_{k=1}^{\infty} C_{2k}^k \frac{1}{2k-1} s^{k-1} = 1/2 \sum_{n=0}^{\infty} C_{2n+2}^{n+1} \frac{1}{2n+1} s^n,$$

де $n = k - 1$. Отже, $a_n = 1/2\, C_{2n+2}^{n+1} \frac{1}{2n+1} = \frac{1}{n+1} C_{2n}^n$. ♠

1.13. Контрольні запитання, задачі і вправи

Контрольні запитання

1) Яка різниця між декартовим квадратом деякої непустої множини A і множиною всіх двохелементних підмножин множини A?

2) Скільки відношень еквівалентності можна побудувати на множині, яка складається з

a) двох, b) трьох, c) чотирьох елементів?

Скільки бінарних відношень можна задати на множині з n елементів?

3) Що називається

a) перестановкою n-елементної множини?

b) сполученням з n елементів по m елементів?

4) Скількома способами можна розмістити три книжки на книжковій полиці?

5) Скільки існує функцій із множини A в множину B, де $|A| = m$, а $|B| = n$?

Задачі і вправи

1. Група студентів налічує 25 осіб. Із них 15 люблять математику, 10 — фізику, 8 не люблять ні математику, ні фізику. Скільки студентів люблять і математику, і фізику?

2. На зборах студентів-відмінників були як студенти другого курсу, так і студенти третього курсу. Всі вони або любителі прози, або любителі поезії. Студентів-хлопців було 16, а любителів прози — 24. Студентів-дівчат було рівно стільки, скільки хлопців любителів прози. Скільки студентів було на зборах?

3. У групі зі 100 студентів англійською мовою володіють 28 осіб, німецькою — 30, французькою — 42, англійською і німецькою — 8, англійською і французькою — 10, німецькою і французькою — 5, а всіма трьома мовами володіють 3 студенти. Скільки студентів не знають ні однієї мови?

4. На кафедрі математики працює сім викладачів. Скількома способами можна скласти комісію із трьох осіб для прийому «хвостів»?

5. У шаховому турнірі брали участь 30 осіб і кожні два шахісти зіграли між собою лише один раз. Скільки партій було зіграно в турнірі?

6. Скільки існує п'ятизначних чисел, у яких кожна наступна цифра
а) менша попередньої; b) більша попередньої.

7. На площині проведено 10 прямих ліній так, що ніякі дві з них не паралельні між собою і ніякі три з них не перетинаються в одній точці. Знайти:

а) число точок перетину цих прямих;

b) число трикутників, які утворюють ці прямі;

c) на скільки частин ділять площину ці прямі.

8. Скільки прямих можна провести через n точок, якщо ніякі три з них не лежать на одній прямій?

9. У корзині знаходиться p білих і q чорних м'ячів. Скількома способами можна викласти ці м'ячі в ряд так, щоб ніякі два чорних м'ячі не були поряд?

10. Скількома способами можна

а) упорядкувати множину $\{1, 2, ..., 2n\}$ так, щоб кожне парне число мало парний номер?

b) розмістити 8 тур на шахівниці так, щоб вони не могли побити одна одну?

c) розмістити 100 книжок на книжковій полиці (звідси буде видно, наскільки необхідним засобом є каталоги в бібліотеках)?

11. У змаганнях з метання списа беруть участь чотири спортсмени $\{A, B, C, D\}$. Скількома способами їх можна розмістити в списку виходів у сектор для метання, якщо спортсмен B не може виходити раніше від спортсмена A?

12. Скільки слів із п'яти літер можна скласти, якщо $X = \{a, b, c, d\}$ і літера a трапляється в слові не більше двох разів, літера b — не більше одного разу і літера c — не більше трьох разів?

13. Нехай $X = \{a, b, c, d\}$ — алфавіт. Слово $p = xyz...uv$ в алфавіті X називається паліндромом, якщо слово $p' = vu...zyx$ дорівнює p. Скільки паліндромів у алфавіті X існує серед слів із п'яти літер?

14. Скільки різних слів можна скласти перестановкою літер у слові «чачача»?

15. Скільки цілих додатних розв'язків має рівняння $x_1 + x_2 + ... + x_m = n$?

16. Обчислити а) $(a + b + c)^2$, $b)$ $(a + b + c)^3$.

17. Знайти коефіцієнт при а) x^5 у розкладі $(1 + x)^7$; $b)$ x^{17} у розкладі $(1 + x^5)^7$.

18. Показати, що сума $C_p^1 + C_p^2 + ... + C_p^{p-1}$ ділиться на p, де p — просте число.

19. Довести, що сума всіх поліноміальних коефіцієнтів дорівнює k^n.

20. Довести малу *теорему Ферма*, тобто, що $a^p - a$ ділиться на p, де a — довільне ціле число, p — просте число.

21. Знайдіть суми

a) $C_n^0 - C_n^1 + C_n^2 - \ldots + (-1)^m C_n^m$;

b) $C_n^0 + C_n^2 + C_n^4 + \ldots$;

c) $C_n^1 - C_n^3 + C_n^5 + \ldots$;

d) $C_{2n}^0 - C_{2n-1}^1 + C_{2n-2}^2 - \ldots + (-1)^n C_n^n$;

f) $(C_n^0)^2 - (C_n^1)^2 + (C_n^2)^2 - \ldots + (-1)^n (C_n^n)^2$;

g) $C_n^1 \sin\alpha + C_n^2 \sin 2\alpha + \ldots + C_n^n \sin n\alpha$;

h) $1 + C_n^1 \cos\alpha + C_n^2 \cos 2\alpha + \ldots + C_n^n \cos n\alpha$.

22. Довести властивості біноміальних коефіцієнтів (f) — (1) з підрозділу 1.11.2.

23. Довести, що в n-елементній множині існує рівно $2^{n-1} - 1$ розбиттів на два класи еквівалентності.

24. Знайти формулу, за якою можна обчислити суму $s(n) = \sum_{k=0}^{n-1} f(k)$, де $s(0) = 0, f(k) = a + bk + ck^2$.

25. У групі вивчається $2n$ предметів. Усі студенти вчаться на 4 і 5. Ніякі два студенти не вчаться однаково, ні про яких двох студентів не можна сказати, що один з них вчиться краще за другого. Показати, що кількість студентів у групі не перевищує C_{2n}^n.

26. Скількома способами можна

а) поселити 8 студентів відповідно в одно-, три- і чотиримісні кімнати?

б) пофарбувати в червоний, синій і зелений колір 2 предмети із шести?

в) розподілити 10 спеціалістів у чотири відділи так, щоб до них потрапили відповідно 1, 2, 3 і 4 спеціалісти?

2 ОСНОВНІ ПОНЯТТЯ ЗАГАЛЬНОЇ АЛГЕБРИ

У цьому розділі розглядаються основні алгебраїчні структури, які знаходять широке застосування в різних галузях науки і техніки. До таких понять відносяться передусім поняття універсальної алгебри, поняття гомоморфізму та ізоморфізму універсальних алгебр, поняття вільної алгебри (вільні напівгрупи, групи, кільця, поля, векторні простори, ґратки, булеві алгебри і т. д.). На закінчення розділу вводяться поняття многоосновної алгебри та її підалгебри, гомоморфізму і ізоморфізму многоосновних алгебр, а також приклади алгебр, які не є універсальними.

2.1. Універсальні алгебри

2.1.1. Загальні відомості

Визначення 31. **Універсальною Ω-алгеброю** (*або просто* **алгеброю**) *називається система* $G = (A, \Omega)$, *яка складається з деякої непустої множини* A **(основна множина алгебри** *або* **носій алгебри)** *і множини, визначених на* A *операцій* $\Omega = \{\omega_1^{k_1}, \omega_2^{k_2}, ..., \omega_n^{k_n}, ...\}$, *фіксованої арності* **(сигнатура алгебри)**, *де* $k_i \in N$, $i = 1, ..., n, ...$. *Операції із множини* Ω *називаються* **основними операціями алгебри.**

Нехай $G = (A, \Omega)$ — довільна алгебра, $\omega \in \Omega$ — n-арна операція і $A' \subseteq A$. Підмножина A' називається *замкнутою* відносно операції ω, якщо для довільних $a_1, ..., a_n$ із A' істинно $\omega(a_1, ..., a_n) \in A'$. Система (A', Ω) називається *підалгеброю алгебри* (A, Ω), якщо $A' \subseteq A$ і A' замкнута відносно довільної основної операції алгебри $G = (A, \Omega)$.

Із визначення підалгебри випливає таке просте твердження.

Теорема 36. *Перетин довільної сукупності підалгебр універсальної алгебри* $G = (A, \Omega)$, *коли він не пустий, буде підалгеброю цієї алгебри.*

Доведення. Нехай $\omega \in \Omega$ — довільна операція арності n і $a_1, a_2, \ldots, a_n \in G_1 \cap G_2 \cap \ldots \cap G_n \cap \ldots$. Тоді, в силу замкнутості операцій із Ω, маємо $\omega(a_1,\ldots, a_n) \in G_i$, $i = 1, 2,\ldots, n,\ldots$. Отже,

$$\omega(a_1,\ldots, a_n) \in G_1 \cap G_2 \cap \ldots \cap G_n \cap \ldots,$$

тобто перетин теж замкнутий відносно операції ω. Тепер справедливість теореми випливає з довільності операції $\omega \in \Omega$. ■

Наслідок 11. *Якщо сигнатура Ω деякої алгебри $G = (A, \Omega)$ включає нульарні операції, то перетин довільної системи підалгебр алгебри G не пустий.*

Дійсно, якщо $\omega \in \Omega$ і $ar(\omega) = 0$, то елемент $\omega \in A$ і, отже, повинен належати всім носіям підалгебр алгебри G. Але тоді перетин цих підалгебр включає, у крайньому випадку, елемент ω. ■

Із теореми 36 випливає, що коли в алгебрі G взята довільна підмножина $D \subseteq A$, то існує однозначно визначена підалгебра $\{D\}$, мінімальна серед підалгебр, які включають множину D. Це буде перетин усіх підалгебр із G, які цілком включають у себе D. Якщо $\{D\} = G$, то D називається **системою твірних** для G. Алгебра $G = \{D\}$ називається **скінченно породженою,** якщо множина D скінченна.

Приклади скінченно породжених алгебр

1. Множина натуральних чисел N породжується множиною $D = \{0, 1\}$ за допомогою операції додавання.

2. Множина чисел N^+ породжується, очевидно, одним елементом $D = \{1\}$ за допомогою операції додавання.

3. Множина цілих чисел Z породжується множиною $D = \{-1, 1\}$ за допомогою операції додавання. ♠

2.1.2. Відношення конгруентності

Нехай R — m-арне відношення, задане на алгебрі $G = (A, \Omega)$. Відношення R називається *стабільним відносно* n-арної операції $\omega \in \Omega$, якщо для довільних $a_{i1}, a_{i2}, \ldots, a_{im} \in A$ $(i = 1, 2, \ldots, n)$ таких, що $(a_{i1}, a_{i2}, \ldots, a_{im}) \in R$, має місце $(\omega(a_{11}, a_{21}, \ldots, a_{n1}),\ldots, \omega(a_{1m}, a_{2m}, \ldots, a_{nm})) \in R$.

Визначення 32. *Відношення R називається* **стабільним на множині** *A, якщо воно стабільне відносно кожної операції із Ω.*

Зокрема, коли R — бінарне відношення, задане на множині A, то воно називається стабільним відносно визначеної на цій множині

n-арної операції ω, якщо для довільних елементів $a_1, a_1', ..., a_n, a_n'$ множини A таких, що

$$a_1 R a_1', \quad a_2 R a_2', \quad \ldots \quad a_n R a_n',$$

має місце

$$\omega(a_1, a_2, ..., a_n) R \omega(a_1', a_2', ..., a_n').$$

Із цих визначень випливає, що тотожно істинні і тотожно хибні відношення на множині A є стабільними відносно довільної основної операції, яка визначена на A.

Визначення 33. *Відношення еквівалентності R, задане на множині A, називається* **конгруентністю,** *якщо R стабільне на A.*

У теоремах 5, 6 розглядалися теоретико-множинні операції над відношеннями еквівалентності. З'ясуємо, які з цих операцій зберігаються для конгруентностей.

Теорема 37. *Якщо R, R_1 — конгруентності на множині A, то*
а) R^{-1} *— конгруентність на множині A;*
б) $R \cap R_1$ *— конгруентність на множині A;*
в) $R * R_1$ *— конгруентність на множині $A \Leftrightarrow R * R_1 = R_1 * R$.*

Доведення. а) Якщо R — конгруентність, то згідно з теоремою 5, $R^{-1} = R$, тобто R^{-1} — конгруентність.

б) З теореми 5 випливає, що $R \cap R_1$ — відношення еквівалентності. Нехай $\omega \in \Omega$ $(ar(\omega) = m)$ і $(a_1, a_1') \in R \cap R_1, ..., (a_m, a_m') \in R \cap R_1$. Тоді, за умовою теореми, маємо $(\omega(a_1, ..., a_m), \omega(a_1', ..., a_m')) \in R$ і $(\omega(a_1, ..., a_m), \omega(a_1', ..., a_m')) \in R_1$. А звідси випливає, що $(\omega(a_1, ..., a_m), \omega(a_1', ..., a_m')) \in R \cap R_1$.

в) З того, що відношення R і R_1 можна міняти місцями, випливає, що $R * R_1$ — відношення еквівалентності, згідно з теоремою 5. Нехай $\omega \in \Omega$, $ar(\omega) = m$ і $(a_1, a_1'), ..., (a_m, a_m') \in R * R_1$. Тоді відповідно до визначення операції добутку відношень, у множині A існують елементи $b_1, ..., b_m$ такі, що $(a_i, b_i) \in R$ і $(b_i, a_i') \in R_1$, $i = 1, 2, ..., m$. В силу того, що R і R_1 — конгруенції, маємо включення $(\omega(a_1, ..., a_m), \omega(b_1, ..., b_m)) \in R$ і $(\omega(b_1, ..., b_m), \omega(a_1', ..., a_m')) \in R_1$. Але звідси випливає, що $(\omega(a_1, ..., a_m), \omega(a_1', ..., a_m')) \in R * R_1$. ■

2.1.3. Гомоморфізми універсальних алгебр

Перейдемо до розгляду основних властивостей універсальних алгебр. Ці властивості пов'язані з поняттями гомоморфізму та ізоморфізму.

Визначення 34. *Універсальні алгебри* $G = (A, \Omega)$ *і* $Q = (B, \Omega')$ *називаються алгебрами одного* **типу,** *якщо між елементами сигнатур* Ω *і* Ω' *можна встановити таку взаємно однозначну відповідність, при якій довільна операція* ω *із* Ω *і відповідна їй операція* ω' *із* Ω' *мають одну й ту саму арність.*

Отже, можна вважати, що в алгебрах одного типу задана одна й та сама сигнатура операцій.

Визначення 35. *Нехай алгебри* $G = (A, \Omega)$ *і* $Q = (B, \Omega)$ *одного типу. Якщо існує відображення* $\varphi : A \to B$ *таке, що для всіх елементів* $a_1,..., a_n$ *із* A *і довільної* n*-арної операції* ω *із* Ω *має місце рівність*

$$\varphi(\omega(a_1,..., a_n) = \omega(\varphi(a_1),..., \varphi(a_n)), \qquad (12)$$

то відображення φ *називається* **гомоморфізмом,** *а алгебра* G *такою, що гомоморфно відображається в алгебру* Q*. Якщо* φ *— взаємно однозначне відображення алгебри* G *на алгебру* Q*, то воно називається* **ізоморфізмом,** *а алгебри* G *і* Q *—* **ізоморфними** ($G \sim Q$)*. Гомоморфізм алгебри* G *в себе* (*тобто* $\varphi : G \to G$) *називається* **ендоморфізмом,** *а гомоморфізм алгебри* G *на себе, називається* **епіморфізмом,** *а коли* φ *ізоморфізм* G *на* G*, то він називається* **автоморфізмом.** *Якщо алгебра* G *ізоморфна деякій підалгебрі алгебри* G'*, то говорять, що алгебра* G **ізоморфно вкладається** *в алгебру* G'*.*

Нехай $\varphi(G)$ означає образ алгебри G при гомоморфізмі φ алгебри G в алгебру G'. Безпосередньо із визначення гомоморфізму алгебр випливають такі прості твердження.

Твердження 1. *Якщо* φ *— гомоморфізм алгебри* G *в алгебру* G' *і* $\varphi(G)$ *— образ алгебри* G *при цьому гомоморфізмі, то* $\varphi(G)$ *— підалгебра алгебри* G'*.*

Доведення. Нехай $G = (A, \Omega)$, $a_1, a_2,..., a_n \in A$ і $\omega \in \Omega$ деяка операція арності n. Покажемо замкнутість множини $\varphi(G)$ відносно основних операцій алгебри G'. Оскільки $\omega(a_1, a_2, ..., a_n) \in G$ і φ — гомоморфізм, то згідно з (12)

$$\varphi(\omega(a_1,..., a_n)) = \omega(\varphi(a_1),..., \varphi(a_n)) \in \varphi(G),$$

оскільки $\varphi(G)$ включає всі образи елементів із G. В силу довільності операції ω гомоморфний образ $\varphi(G)$ алгебри G є підалгеброю алгебри G'. ■

Твердження 2. *Добуток гомоморфізмів алгебр теж є гомоморфізмом.*

Доведення пропонується читачеві як вправа (див. вправу 1 у кінці розділу).

Існує деякий загальний метод огляду всіх гомоморфних образів універсальних алгебр.

Нехай $G = (A, \Omega)$ — деяка універсальна алгебра, R — конгруенція на G і $A_1, A_2,..., A_n$ — класи розбиття множини A за відношенням R. В силу того, що R — конгруенція, клас B елемента $\omega(a_1, ..., a_n)$, де $a_i \in A$, $\omega \in \Omega$, $ar(\omega) = n$, визначається однозначно і не залежить від вибору елементів $a_1, a_2,..., a_n$ у класах $A_1, A_2,..., A_n$. Це дає можливість визначити операцію ω на фактор-множині G/R, припускаючи $\omega(A_1,..., A_n) = B$, якщо $\omega(a_1, a_2, ..., a_n) = b$, де $a_1 \in A_1$, $a_2 \in A_2$, ..., $a_n \in A_n$, $b \in B$. Оскільки сказане має місце для довільної операції $\omega \in \Omega$, то G/R стає універсальною алгеброю тієї ж сигнатури, що й алгебра G. Ця алгебра називається **фактор-алгеброю** алгебри G по конгруенції R. Оскільки алгебри G і G/R одного типу, то можна говорити про гомоморфізм чи ізоморфізм алгебри G на G/R.

Розглянемо відображення $\varphi : G \to G/R$, яке ставить у відповідність довільному елементу $a \in A$ клас еквівалентності, котрий включає цей елемент. Зауважимо, що φ є відображенням *на*. Покажемо, що воно є гомоморфізмом.

Нехай

$$(a_1, a_2,..., a_n) \in G, \omega \in \Omega, ar(\omega) = n.$$

Тоді

$$\varphi(\omega(a_1, a_2,..., a_n)) = \omega(\varphi(a_1), \varphi(a_2),..., \varphi(a_n)) = \omega(A_1, A_2,..., A_n)$$

за визначенням операції ω на G/R, тобто φ — гомоморфізм, який називається **натуральним гомоморфізмом.**

Теорема 38. (*Теорема про гомоморфізми*). *Якщо f — гомоморфізм алгебри $G = (A, \Omega)$ на алгебру $G' = (B, \Omega)$, то на алгебрі G існує така конгруенція R, що алгебра G' ізоморфна фактор-алгебрі G/R і коли g — цей ізоморфізм, то відображення $f * g$ збігається з натуральним гомоморфізмом φ алгебри G на G/R.*

Доведення. Визначимо відношення R на A таким чином: $(a, b) \in R \Leftrightarrow$ $\Leftrightarrow f(a) = f(b)$. Із теореми 11 випливає, що R — еквівалентність, а відображення $f * g$ — взаємно однозначне. Покажемо, що R — конгруентність, а $f * g$ — ізоморфізм.

Нехай $f(a_i) = f(b_i)$ для деяких a_i, $b_i \in G$, $i = 1, 2, ..., n$ і $\omega \in \Omega$, $ar(\omega) = n$. Тоді

$$f(\omega(a_1, a_2, ..., a_n)) = \omega(f(a_1), f(a_2), ..., f(a_n)) =$$
$$= \omega(f(b_1), f(b_2), ..., f(b_n)) = f(\omega(b_1, b_2, ..., b_n)),$$

тобто R — конгруентність і фактор-алгебра G/R існує.

Нехай a_i' — довільні елементи із G'. Задаючи $g(a_i') = A_i$, $i = 1, 2, ..., n$, і вибираючи елементи a_i із A_i такі, що $f(a_i) = a_i'$, одержуємо

$$f(\omega(a_1, .., a_n)) = \omega(f(a_1), ..., f(a_n)) = \omega(a_i', ..., a_n').$$

Оскільки $\omega(a_1, ..., a_n) \in \omega(A_1, ..., A_n)$, то

$$g(\omega(a_1', ..., a_n') = \omega(A_1 ..., A_n) = \omega(g(a_1'), ..., g(a_n')). \blacksquare$$

Приклад. Нехай $G = (D^+, \Omega)$ — алгебра додатних дійсних чисел, тобто $D^+ = \{x \in D | x > 0\}$, а Ω включає бінарну операцію множення, унарну операцію взяття оберненого елемента до заданого відносно операції множення і нульарну операцію, яка фіксує одиничний елемент — 1. Алгебра $G' = (D, \Omega')$, де D — множина всіх дійсних чисел, а Ω' включає бінарну операцію додавання, унарний мінус і нульарну операцію 0.

Відображення $f = lg$ — десятковий логарифм є ізоморфізмом алгебри G на G'. Дійсно,

a) $lg(a \cdot a') = lg\, a + lg\, a'$;

b) $lg(1/a) = lg(1) - lg(a) = 0 - lg\, a = -lg\, a$;

c) $lg\, 1 = lg(a \cdot (1/a)) = lg\, a - lg\, a = 0$.

Таким чином, операції при цьому відображенні зберігаються відповідно до визначення ізоморфізму.

d) Покажемо тепер, що це відображення взаємно однозначне. Нехай $a \neq a'$, але $lg\, a = lg\, a'$. Тоді

$$lg\, a - lg\, a' = lg(a/a') = 0$$

і, отже, $a/a' = 1$ і $a = a'$. А це є хибним по відношенню до нашого припущення. Обернене твердження доводиться аналогічно.

Згідно з *a*) — *d*) робимо висновок, що дане відображення $f = lg$ — ізоморфізм. ♠

2.1.4. Алгебра (мова) термів

Для задання деякої формальної мови необхідно задати її алфавіт і правила, за якими будуються слова із символів цього алфавіту.

Нагадаємо, що довільна сукупність попарно різних символів

$$X = \{x_1, x_2, ..., x_n, ...\}$$

називається **алфавітом.** Символи алфавіту часто ще називають *літерами.* Найпростіший приклад мови — це *мова слів* у алфавіті X. Скінченна послідовність $p = x_{i1} \dots x_{ik}$ літер складає **слово в алфавіті** X, якщо $x_{ij} \in X$, $j = 1, 2,..., k$. При цьому літери x_{ij} називають літерами, що складають слово p, а довжиною $l(p)$ слова p — число літер, що його складають. Серед слів, довжина яких виражається цілим додатним числом, розглядається слово нульової довжини, яке за визначенням не має жодного символу і називається *пустим словом.* Для позначення цього слова вводиться спеціальний символ e.

Зауважимо, що коли p, q — деякі слова в алфавіті X, тобто послідовності $x_{i1} \dots x_{ik}$ і $x_{j1} \dots x_{jl}$ відповідно, то послідовність $pq = x_{i1} \dots x_{ik} x_{j1} \dots x_{jl}$, одержана в результаті приписування слова q праворуч до слова p, теж буде, очевидно, словом в алфавіті X. Це випливає безпосередньо з визначення слова. Отже, таке сполучення слів у алфавіті X можна розглядати як операцію на множині слів алфавіту X. Ця операція носить назву **конкатенації** або **добутку слів.**

Умовимося позначати через $F(X)$ сукупність усіх слів у алфавіті X, включаючи і пусте слово e. Множину $F(X)$ будемо називати **мовою слів.**

Для задання функцій і операцій, а також для вивчення їх властивостей користуються особливою формальною мовою — **мовою термів.**

Мова термів визначається за допомогою індукції таким чином.

Алфавіт мови термів складають символи, розбиті на три групи: T_0, Ω, $\{(,)\}$. Символи із T_0 — символи першої групи — називаються **предметними.** Такими символами служать літери a, b, x, y,... або ті ж самі літери з індексами. Символи із Ω — символи другої групи — називаються **функціональними.** Це літери з верхніми і, можливо, нижніми індексами:

$$f_2^3, g_2^5, f_5, \dots .$$

Верхній індекс n ($n \geq 1$) вказує (як і раніше) арність функціонального символу. Якщо його немає, то функціональний символ вважається унарним. І символи третьої групи — це **ліва, права** дужки і кома.

Визначення 36. *Термами називаються слова, побудовані за такими правилами:*

1) усі символи із T_0 є термами;

2) якщо $t_1, \dots, t_n$ — терми, то слово вигляду $f^n(t_1, \dots, t_n)$ — терм ($f^n \in \Omega$, $n \geq 1$);

3) термами є ті і тільки ті слова, про які йдеться в пунктах 1, 2.

Множина всіх термів у алфавіті X сигнатури Ω позначається $T(\Omega, X)$. Якщо $t = \omega(t_1,..., t_n) \in T(\Omega, X)$, то терми $t_1,..., t_n$ називаються безпосередніми підтермами терма t. Транзитивне замикання відношення «безпосередній підтерм» називається відношенням «підтерм терма».

Множину термів $T(\Omega, X)$ можна розглядати як універсальну Ω-алгебру, якщо визначитися з операціями нульової арності, оскільки при визначенні $T(\Omega, X)$ вважалося, що $ar(\omega) \geq 1$ для довільної операції $\omega \in \Omega$.

Нехай $X = \{x_1, x_2, \ldots, x_n,\ldots\}$ — деякий алфавіт, а Ω — множина операцій. Представимо множину Ω у вигляді $\Omega = \Omega' \cup \Omega_0$, де Ω_0 означає множину нульарних операцій, а $\Omega' = \Omega \backslash \Omega_0$. Позначимо через $T(\Omega', X)$ множину термів, у якої $T_0 = X \cup \Omega_0$, а множиною функціональних символів є Ω'. Тепер $T(\Omega', X)$ можна розглядати як універсальну Ω'-алгебру з системою операцій Ω'. Надалі, щоб не ускладнювати викладки і не вводити додаткові позначення, будемо алгебру $T(\Omega', X)$ позначати, як і раніше, $T(\Omega, X)$, маючи на увазі описаний перехід, і називати її **алгеброю термів.**

Нехай маємо довільну алгебру $G = (A, \Omega)$ (алгебра G може збігатися з $T(\Omega, X)$) і алгебру $T(\Omega, X)$. Якщо $\Omega_0 \neq \varnothing$, то нехай $a(f)$ означає елемент із A, який відповідає нульарній операції f із Ω_0. Розглянемо відображення φ: $T_0 \to A$ таке, що $\varphi(f) = a(f)$ для $f \in \Omega_0$. Відображення φ можна продовжити на всю алгебру $T(\Omega, X)$, якщо для p_1, $p_2,\ldots, p_n \in T(\Omega, X)$ і $f^n \in \Omega$ $(n \geq 1)$ покласти $\varphi(f(p_1,\ldots, p_n)) = f(\varphi(p_1), \ldots, \varphi(p_n))$.

Відображення φ називається **інтерпретацією** алгебри $T(\Omega, X)$ на алгебрі G.

Говорять, що в алгебрі G виконується **тотожне співвідношення** (або просто **тотожність**) $p_1 = p_2$, коли $\varphi(p_1) = \varphi(p_2)$ в G при довільній інтерпретації φ. Якщо дана множина співвідношень Eq, то сукупність алгебр сигнатури Ω, в яких виконуються всі співвідношення із Eq, складають клас алгебр, який позначатиметься через $K(\Omega, Eq)$. Співвідношення із множини Eq називаються **тотожними.**

Нехай $p_1 = p_2 \in Eq$, $p, q \in T(\Omega, X)$ і терм p одержаний із терма q в результаті підстановки p_1 замість деякого входження терма p_2 в терм q. У цьому випадку говорять, що терм p **безпосередньо виводиться** з терма q за допомогою тотожного співвідношення $p_1 = p_2$ із Eq. Транзитивне замикання відношення безпосереднього виведення називається просто виведенням. Терми q_1 і q_2 називаються еквівалентними відносно Eq, якщо один із них виводиться з другого (p_1 Req q_2). Відношення Req буде, очевидно, еквівалентністю і навіть, що легко перевірити, конгруентністю.

Визначення 37. *Нехай* $T(\Omega, X, Eq)$ *означає фактор-алгебру* $T(\Omega, X)/Req$. *Алгебра* $T(\Omega, X, Eq)$ *називається* **вільною алгеброю** *класу* $K(\Omega, Eq)$, *а множина* X — *її* **системою вільних твірних (базисом).** *Алгебра* $T(\Omega, X, Eq)$ *називається* **спадково вільною**, *якщо довільна її підалгебра теж є вільною алгеброю.*

Із визначення $T(\Omega, X, Eq)$ випливає, що $T(\Omega, X, Eq) \in K(\Omega, Eq)$. Важливість поняття вільної алгебри випливає із такого твердження.

Теорема 39. *Довільна алгебра* G *із класу* $K(\Omega, Eq)$ *є гомоморфним образом вільної алгебри* $T(\Omega, X, Eq)$ *цього класу* [21].

2.1.5. Похідні операції і скінченні алгебри

Крім основних операцій алгебри, часто розглядаються похідні операції, число яких може бути нескінченним.

Нехай $\omega(x_1,..., x_n) \in T(\Omega, X, Eq)$, $ar(\omega) > 1$ і $x_1,..., x_n \in X$ — вільні твірні. Замінимо k $(0 \le k \le n)$ із вільних твірних, скажімо $x_{n-k+1},..., x_n$, деякими фіксованими елементами $a_1, a_2,..., a_k$ із $T(\Omega, X, Eq)$. Вираз $\omega(x_1,..., x_{n-1}, a_1,..., a_k)$, який при цьому буде одержаний, визначає на $T(\Omega, X, Eq)$ $(n - k)$-арну операцію: системі елементів $b_1,..., b_{n-k} \in T(\Omega, X, Eq)$ ця операція співставляє однозначно визначений елемент $\omega(b_1,..., b_{n-k}, a_1,..., a_k)$.

Усі операції, які можуть бути одержані в $T(\Omega, X, Eq)$ таким шляхом, називаються *похідними операціями алгебри* $T(\Omega, X, Eq)$. Похідна операція називається головною, якщо $k = 0$, тобто коли в термі $\omega(x_1,..., x_n)$ не виконується ніякої заміни вільних твірних. Зауважимо, що коли сигнатура Ω включає похідні операції, то таке поняття, як підалгебра даної алгебри втрачає сенс, оскільки підалгебра не обов'язково повинна включати елементи $a_1,..., a_k$, які беруть участь у визначенні похідної операції. Аналогічне зауваження стосується і поняття гомоморфізму. Всі ці складності зникають, якщо вважати, що кожна операція сигнатури Ω головна. Надалі будемо вважати, що сигнатура Ω включає тільки головні операції і не включає ніяких похідних операцій. На завершення цього підрозділу коротко зупинимось на скінченних алгебрах.

Алгебра називається **скінченною,** якщо її носій має скінченне число елементів. Якщо алгебра G — скінченна і складається із n елементів, то її називають **алгеброю *n*-го порядку.**

Скінченну алгебру G n-го порядку в деяких випадках зручно задавати у вигляді прямокутних таблиць, які називаються **таблицями Келі.** Кожна така таблиця відповідає деякій операції ω із Ω і будується за такими правилами.

Якщо ω — k-арна операція із Ω і $\omega(a_1, a_2, ..., a_k) = b$, то в таблиці послідовності $a_1, a_2, \ldots, a_k, b$ відповідає рядок $a_1, a_2, ..., a_k, b$. Іншими словами, кожній операції ω алгебри G відповідає рядок елементів $a_1, a_2, \ldots, a_k, b$ такий, що $\omega(a_1, a_2, \ldots, a_k) = b$.

ω	1	2	...	k	$k+1$
1	a_{11}	a_{12}	...	a_{1k}	b_1
2	a_{21}	a_{22}	...	a_{2k}	b_2
...	...	...	...	...	...
...	...	...	...	...	...
...	...	...	...	...	...
s	a_{s1}	a_{s2}	...	a_{sk}	b_q,

де $q \le n$ а $s = n^k$.

Якщо ж ω — бінарна операція, то ця таблиця задається квадратною таблицею вигляду

ω	a_1	a_2	...	a_n
a_1	a_{11}	a_{12}	...	a_{1n}
a_2	a_{21}	a_{22}	...	a_{2n}
...	...	...	...	...
...	...	...	...	...
...	...	...	...	...
a_n	a_{n1}	a_{n2}	...	a_{nn},

в якій на перетині i-го рядка і j-го стовпчика стоїть значення операції $\omega(a_i, a_j)$. Коли операція унарна, то це буде просто рядок, який складається із n елементів, i-й елемент якого дорівнює значенню $\omega(a_i)$, $i, j = 1, 2, ..., n$.

Приклади відповідних представлень скінченних алгебр будуть розглядатися більш детально в наступних розділах, а тому перейдемо до розгляду основних широко відомих алгебр, які відіграють важливу роль як у теорії, так і на практиці.

2.2. Вільні алгебри та їх основні властивості

2.2.1. Абсолютно вільні алгебри

Визначення 38. *Алгебра* $T(\Omega, X, Eq)$ *називається* **абсолютно Вільною**, *коли* $Eq = \varnothing$, *тобто тоді і тільки тоді, коли* $T(\Omega, X, Eq) \sim T(\Omega, X)$.

Із визначення алгебри термів $T(\Omega, X)$ випливає, що множина X служить для цієї алгебри системою твірних, і що алгебри термів над рівнопотужними алфавітами ізоморфні. Оскільки в $T(\Omega, X)$ не виконується ні одне нетривіальне співвідношення, то довільний елемент із $T(\Omega, X)$ єдиним способом представляється через елементи алфавіту X і символи нульарних операцій із Ω.

Сенс терміна «абсолютно вільна» для алгебри $T(\Omega, X)$ у класі всіх алгебр сигнатури Ω випливає із такого твердження.

Твердження 3. *Довільне відображення f множини твірних X алгебри термів $T(\Omega, X)$ у довільну алгебру G того самого типу, що й алгебра $T(\Omega, X)$, можна продовжити єдиним способом до гомоморфізму φ алгебри $T(\Omega, X)$ на алгебру G.*

Дійсно, припустимо для всіх x із X – $\varphi(x) = f(x)$ і для всіх нульарних операцій ω із Ω – $\varphi(\omega) = \omega$. Якщо для $\omega \in \Omega$, $ar(\omega) = n \geq 1$, образи термів $t_1, t_2, \ldots, t_n$ уже визначені, то, припускаючи $\varphi(\omega(t_1, t_2, \ldots, t_n)) = \omega(\varphi(t_1), \varphi(t_1), \varphi(t_2), \ldots, \varphi(t_n), \ldots, \varphi(t_n))$, одержимо, що відображення φ — гомоморфізм.

Єдиність такого відображення φ випливає з єдиності представлення термів у алгебрі $T(\Omega, X)$.

2.2.2. Вільні групоїди

Визначення 39. *Абсолютно вільна алгебра $T(\Omega, X)$ називається* **вільним групоїдом**, *якщо $Eq = \varnothing$, а Ω включає єдину бінарну операцію, що називається множенням слів у алфавіті X.*

Словом, у цій алгебрі буде довільна скінченна упорядкована система елементів із X

$$x_{i1}x_{i2}\ldots x_{in}\ (n > 1),$$

де x_{ij} $(j = 1, 2, \ldots, n)$ не обов'язково всі різні, причому в цій алгебрі заданий розподіл дужок, які визначають порядок виконання операції множення. При цьому кожний символ $x_{ij} \in X$ вважається взятим у дужки, а множення двох слів означає, що задані слова беруться у дужки і пишуться одне за другим. Наприклад, якщо

$$p = (x_{i1})(x_{i2}),\ q = (x_{i1})((x_{i2})(x_{i1})),$$

то

$$pq = ((x_{i1})(x_{i2}))((x_{i1})((x_{i2})(x_{i1}))).$$

2.2.3. Вільні напівгрупи

Якщо Ω включає єдину бінарну операцію — множення, а Eq — єдине тотожне співвідношення — асоціативність множення, то $T(\Omega, X, Eq)$ називається **вільною напівгрупою.** Коли існує елемент $e \in T(\Omega, X, Eq)$ і Eq включає тотожності $pe = ep = p$ для всіх $p \in T(\Omega, X, Eq)$, то $T(\Omega, X, Eq)$ називається **вільною напівгрупою з одиницею** або **вільним моноїдом.**

Приклади напівгруп

1. Прикладом вільної напівгрупи $T(\Omega, X)$ може служити введена раніше мова $F(X)$ — мова слів у алфавіті X.

Нехай $p, q \in T(\Omega, X)$. Роль операції множення в цій напівгрупі відіграє операція конкатенації, яка, як легко зрозуміти, задовольняє закон асоціативності. Якщо пусте слово e належить множині $T(\Omega, X)$, то ця напівгрупа буде мати одиницю. Підтерм p терма q називається **підсловом** слова q. Якщо p — підслово слова q, то q можна представити у вигляді добутку

$$q = p'pp'', \tag{13}$$

де p', p'' — підходящі (можливо пусті) слова із $T(\Omega, X)$. Якщо в розкладі (13) слово p' найменшої довжини, яка можлива, то говорять про перше входження слова p в слово q. Аналогічно можна говорити про друге, третє і т.д. входження p в q. Нехай розклад (13) відповідає k-му входженню p в q, тоді слово $q = p'q'p''$ називається словом, одержаним у результаті підстановки слова q' замість k-го входження p в q.

Оскільки формальні мови відіграють важливу роль у побудові мов програмування, то виникає питання про потужність мови слів над деяким скінченним алфавітом і, зокрема, про потужність множини всіх мов над цим алфавітом.

Твердження 4. *Множина всіх слів мови $F(X)$ над скінченним алфавітом X є множина зліченна.*

Доведення випливає з того, що цю мову можна представити у вигляді об'єднання

$$F(X) = \{e\} \cup L_1 \cup L_2 \cup \ldots \cup L_n \cup \ldots,$$

де L_k — множина всіх слів із $F(X)$, довжина яких дорівнює k, $k = 1, 2, \ldots$. Справедливість даного твердження випливає з теореми 19. ■

Твердження 5. *Множина всіх мов над скінченним алфавітом X є незліченною множиною* [9].

Дійсно, множина всіх мов у алфавіті X є булеаном множини X, і справедливість даного твердження випливає з теореми про булеан. ■

2. Напівгрупою є **сукупність усіх бінарних відношень,** заданих на деякій множині A, відносно операції множення відношень, оскільки ця операція асоціативна (теорема 3). Це буде напівгрупа з одиницею. Роль одиниці відіграє відношення тотожності i_A.

Сукупність усіх відношень еквівалентності, заданих на множині A, не буде напівгрупою, оскільки ця сукупність у загальному випадку не замкнута відносно операції множення відношень (теорема 5). З цієї причини і сукупність усіх конгруентностей на множині A не буде напівгрупою.

3. **Перетворенням множини** A називається довільне відображення f множини A в себе, тобто $f: A \to A$. Очевидно, що коли f і g — перетворення множини A, то $f * g$ — теж перетворення множини (див. вправи), тобто множина всіх перетворень множини A замкнута відносно операції множення перетворень. Як було показано вище, операція множення відображень асоціативна, і, отже, **сукупність усіх перетворень множини** A складає напівгрупу, яка називається **симетричною напівгрупою** на множині A. Важливість цієї напівгрупи висвітлює

Твердження 6. *Довільна напівгрупа G ізоморфно вкладається в симетричну напівгрупу на деякій множині A* [21, 14].

Якщо множина $A = \{1, 2,..., n\}$ — скінченна, то симетрична напівгрупа на множині A теж скінченна. У цьому випадку довільне перетворення f задається таблицею відповідностей

$$f = \begin{pmatrix} 1 & 2 & \dots & n \\ a_1 & a_2 & \dots & a_n \end{pmatrix},$$

де кожний стовпчик означає $f(i) = a_i$, $a_i \in A$, $i = 1, 2,..., n$.

4. Нехай G — напівгрупа і a — довільний її елемент. Асоціативність множення дозволяє звичайним способом визначити додатні степені a^n елемента a, $n = 1, 2,...$, причому

$$a^k a^l = a^{k+l}, \ (a^k)^l = a^{kl}.$$

Звідси видно, що додатні степені елемента a складають піднапівгрупу напівгрупи G. Ця напівгрупа комутативна і називається **циклічною піднапівгрупою** елемента a. ♠

2.2.4. Вільні комутативні напівгрупи

Вільна напівгрупа $T(\Omega, X, Eq)$ називається **вільною комутативною напівгрупою,** якщо множина Eq, крім асоціативності, включає тотожне співвідношення комутативності множення слів.

Тотожні співвідношення дозволяють записати довільне слово із $T(\Omega, X, Eq)$ у вигляді

$$x_1^{n_1} x_2^{n_2} \ldots x_k^{n_k},$$

де $x_i \in X$, n_i — цілі невід'ємні числа. Під виразом x^n розуміють слово $xx \ldots x$ довжини n при $n > 0$, а при $n = 0 - x^n = e$.

Прикладом вільної комутативної напівгрупи відносно операції додавання може служити множина N^+, яка, як зазначалося вище, породжується єдиною вільною твірною -1, тобто $X = \{1\}$. Операція додавання на N^+ асоціативно-комутативна.

2.2.5. Вільні групи

Нехай Ω включає бінарну операцію конкатенації слів, унарну операцію взяття оберненого елемента і нульарну операцію, яка фіксує одиничний елемент. Наявність операції взяття оберненого елемента дозволяє однозначно співставити множині X множину елементів X^{-1}. Символи x із X і x^{-1} із X^{-1} називаються *взаємно оберненими.* Слово в алфавіті $\overline{X} = X \cup X^{-1}$ називається *нескорочуваним*, якщо воно не має жодної пари взаємно обернених символів, які стоять поруч. Довільне слово можна перетворити до нескорочуваного слова в результаті послідовного викреслювання пар взаємно обернених символів, які стоять поруч. Очевидно, що пусте слово e буде нескорочуваним. Таке перетворення слів називається їх *приведенням.*

Визначення 40. *Якщо Eq включає співвідношення асоціативності, співвідношення скорочення* $(x \cdot x^{-1} = x^{-x} \cdot x = e)$ *і співвідношення, пов'язані з одиницею* $(x \cdot e = e \cdot x = x)$, *то* $T(\Omega, X, Eq)$ *називається* **вільною групою.** *Одиницею групи* $T(\Omega, X, Eq)$ *є пусте слово* e.

Тотожні співвідношення дозволяють записати довільне слово із $T(\Omega, X, Eq)$ у вигляді нескорочуваного слова

$$x_{i_1}^{n_1} x_{i_2}^{n_2} \ldots x_{i_k}^{n_k},$$

де $x_{ij} \in \overline{X}$, n_i — ціле число, x_{ij} не обов'язково всі різні, $j = 1,\ldots, k$, а x^n — слово довжини n, яке визначається таким чином:

$$x^n = \begin{cases} \underbrace{xx...x}_{n \text{ разів}}, & \text{якщо } n > 0; \\ e, & \text{якщо } n = 0; \\ \underbrace{x^{-1}x^{-1}...x^{-1}}_{n \text{ разів}}, & \text{якщо } n < 0. \end{cases}$$

Нехай G — група, H — деяка її підгрупа і a — довільний елемент із G. Множина $aH = \{ah | h \in H\}$ називається *лівим суміжним класом групи G по підгрупі H*, заданим елементом a. Зрозуміло, що $a \in aH$, оскільки $e \in H$, і якщо $b \in aH$, то $bH = aH$. Дійсно, $b \in aH$ означає, що $b = ah$, де $h \in H$. Але тоді для довільних h_1, $h_2 \in H$ маємо $bh_1 = (ah)h_1 = a(hh_1) \in aH$, тобто $bH \subseteq aH$. З другого боку,

$$ah_2 = (bh^{-1})h_2 = b(h^{-1}h_2) \in bH,$$

тобто $aH \subseteq bH$. Отже, $aH = bH$.

Звідси випливає, що ліві суміжні класи H, a_1H, a_2H,..., a_nH,... є класами розбиття групи G. В силу теореми 4 підгрупа H задає на G відношення еквівалентності R : $aRb \Leftrightarrow aH = bH$.

Задання групи G у вигляді об'єднання класів $H \cup a_1H \cup a_2 \cup a_3H \cup ... \cup a_nH \cup ...$ називається лівостороннім розкладом групи G по підгрупі H. Аналогічно будується і правосторонній розклад групи G по підгрупі H. Правосторонній і лівосторонній розклади групи складаються з одного й того самого числа класів. У цьому легко переконатися, задаючи відображення f: $G \to G$ так, що $f(a) = a^{-1}$.

Якщо ж число суміжних класів у розкладі групи G по підгрупі H скінченно, то підгрупа називається підгрупою **скінченного індексу** а число класів — **індексом** підгрупи H у групі G.

Підгрупа H називається **нормальним дільником** або **інваріантною підгрупою** групи G, якщо лівосторонній розклад G по H збігається з правостороннім розкладом G по H. Інакше кажучи, для довільного $a \in G$ має місце рівність $aH = Ha$ або $aHa^{-1} = H$. Очевидно, що нормальними дільниками довільної групи буде одинична підгрупа, яка складається лише з одиниці групи, і сама група. Ці нормальні дільники називаються *тривіальними*.

Нехай G і G' — групи, e і e' — їх одиничні елементи відповідно, а h — гомоморфізм групи G в групу G'. Множина $\{a \in G | h(a) = e'\}$ називається **ядром гомоморфізму** h і позначається символом $ker(h)$. Неважко довести, що $ker(h)$ — нормальний дільник групи G (див. вправу 15).

Якщо група скінченна, то всі її підгрупи теж будуть скінченними, а число її елементів називатиметься *порядком групи*. Нехай G —

скінченна група n-го порядку і H — її підгрупа k-го порядку. Тоді в розкладі G по H довільний суміжний клас складається в точності з k елементів і, отже, $n = kj$. Звідси випливає

Теорема 40. (*Теорема Лагранжа*) *Порядок і індекс довільної підгрупи скінченної групи є дільниками порядку групи.*

Якщо G — група і $a \in G$, то припустимо $a^0 = e$ і $a^{-n} = (a^{-1})^n$, а додатні степені елемента a визначалися вище для циклічної напівгрупи елемента a.

Сукупність елементів вигляду a^n, де $n \in Z$, називається *циклічною підгрупою* групи G. Степені елемента a не обов'язково повинні бути різними. Елемент a групи G називається **елементом скінченного порядку**, якщо існують такі цілі числа k і l, що $a^k = a^l$, тобто $a^{k-l} = e$. Найменший показник серед усіх таких показників елемента a називається *порядком елемента a*. Якщо таких чисел k і l не існує, то a має нескінченний порядок.

Визначення 41. *Група, усі елементи якої мають нескінченний порядок, називається* **групою без кручень.** *Група, усі елементи якої мають скінченні порядки* (*не обов'язково обмежені в сукупності*), *називається* **періодичною групою.**

Якщо a — елемент скінченного порядку, то $a^0 = e$, a, a^2,..., a^{n-1} будуть різними елементами групи. Якщо дано a^k, де $k > n$, то $k = qn + r$, $0 \le r < n$, і $a^{qn+r} = (a^n)^q \cdot a^r = a^r$. Отже, скінченний порядок елемента збігається з порядком його циклічної групи.

Наслідок 12 (*Наслідок 1 теореми Лагранжа*). *Порядок довільного елемента скінченної групи є дільником порядку групи.*

Наслідок 13 (*Наслідок 2 теореми Лагранжа*). *Довільна скінченна група, порядок якої є простим числом, буде циклічною.*

Дійсно, з огляду на простоту порядку групи, вона повинна збігатися з циклічною підгрупою, породженою довільним її елементом відмінним від одиниці. ■

Група називається *простою*, якщо вона не має нетривіальних нормальних дільників.

Якщо група H — нормальний дільник групи G, то множина суміжних класів складає групу. Дійсно, якщо H, a_1H, a_2H,... — розбиття групи G і для довільного елемента a із G мають місце рівності $aH = Ha$ і $H \cdot H = H$, то неважко довести, що:

1) $(aH \cdot bH) \cdot cH = aH \cdot (bH \cdot cH)$;

2) $aH \cdot H = H \cdot aH = aH = Ha$, тобто H відіграє роль одиниці;
3) $aH \cdot (a^{-1})H = (aH \cdot (a^{-1}) \cdot H = H \cdot H = H$;
4) $aH \cdot bH = (Ha) \cdot (bH) = (H(ab)) \cdot H = abH \cdot H = Hab$.

Іншими словами, довільний нормальний дільник H групи G задає деяке відношення конгруентності R на G. Фактор-група по цьому відношенню конгруентності позначається G/H і називається **фактор-групою групи** G по нормальному дільнику H.

Якщо група G — скінченна і H — її нормальний дільник, то отримуємо

Наслідок 14 (*Наслідок 3 теореми Лагранжа*). *Порядок групи G/H дорівнює індексу підгрупи H у групі G і є дільником порядку групи G.*

Приклад. Розглянемо приклад некомутативної групи, яка відіграє важливу роль у теорії груп, — групу підстановок деякої скінченної множини $M = \{1, 2,\ldots, n\}$. Вище було введено поняття перетворення множини (див. приклади напівгруп). Окремим випадком перетворення є **підстановка**, тобто взаємно однозначне відображення множини M на себе.

З попереднього відомо, що добуток відображень — це їх послідовне виконання і що ця операція асоціативна.

Сукупність усіх підстановок множини M складає групу. Дійсно, роль одиниці в цій групі відіграє тотожна підстановка, яка залишає на місці кожний елемент із M. З іншого боку, якщо елемент a переходить в елемент $f(a)$, то $f^{-1}(f(a)) = a$ і f^{-1} теж буде підстановкою в силу взаємної однозначності f. f^{-1} буде відігравати роль оберненого елемента для f. Отже, усі підстановки множини M складають групу, яка називається **симетричною групою на множині** M.

Якщо множина M скінченна і складається із n елементів, то симетрична група на M, яка називається **симетричною групою n-го степеня**, буде скінченною і матиме порядок $n!$ (див. вправу 14 у кінці розділу). Підстановки скінченних множин задаються у вигляді таблиць відповідностей. Наприклад, якщо $M = \{1, 2, 3, 4, 5, 6\}$, то підстановка f множини M, рівна $\{4, 3, 1, 5, 2, 6\}$, матиме вигляд

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 4 & 3 & 1 & 5 & 2 & 6 \end{pmatrix}.$$

Множення підстановок $f \cdot f_1$ являє собою суперпозицію відображень, тобто $f \cdot f_1(i) = f_1(f(i))$ і записується таким чином:

$$f = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 4 & 3 & 1 & 5 & 2 & 6 \end{pmatrix} \cdot f_1 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 6 & 3 & 5 & 2 & 4 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 2 & 5 & 6 & 4 & 3 & 1 \end{pmatrix}.$$

Нехай маємо підстановки f_1, f_2, f_3, задані такими наведеними нижче таблицями. Для прикладу покажемо, що $(f_1 \cdot f_2) \cdot f_3 = f_1 \cdot (f_2 \cdot f_3)$:

$$f_1 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 4 & 3 & 5 & 2 & 6 & 1 \end{pmatrix}, \quad f_2 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 3 & 5 & 2 & 6 & 1 & 4 \end{pmatrix},$$

$$f_3 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 6 & 3 & 5 & 1 & 4 & 2 \end{pmatrix}.$$

Тоді

$$(f_1 \cdot f_2) \cdot f_3 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 2 & 3 & 6 & 4 & 1 & 5 \end{pmatrix}, \quad f_1 \cdot (f_2 \cdot f_3) = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 2 & 3 & 6 & 4 & 1 & 5 \end{pmatrix}.$$

Розглянемо приклад задання групи G всіх підстановок множини $M = \{1, 2, 3\}$ за допомогою таблиці Келі. У цьому випадку група G — скінченна алгебра 6-го порядку і її елементами є підстановки

$$f_1 = \begin{pmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \end{pmatrix}, \quad f_2 = \begin{pmatrix} 1 & 2 & 3 \\ 3 & 1 & 2 \end{pmatrix}, \quad f_3 = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 3 & 1 \end{pmatrix},$$

$$f_4 = \begin{pmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \end{pmatrix}, \quad f_5 = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 1 & 3 \end{pmatrix}, \quad f_6 = \begin{pmatrix} 1 & 2 & 3 \\ 1 & 3 & 2 \end{pmatrix}.$$

Таблиця множення цієї алгебри має вигляд

$\cdot$	1	2	3	4	5	6
1	1	2	3	4	5	6
2	2	3	1	6	4	5
3	3	1	2	5	6	4
4	4	5	6	1	2	3
5	5	6	4	3	1	2
6	6	4	5	2	3	1,

де $h(f_i) = i$, $i = 1, 2, 3, 4, 5, 6$.

Таблицю операції взяття оберненого елемента представлено в таблиці множення, тому наводити її не має сенсу ♠.

2.2.6. Вільні абелеві групи

Визначення 42. *Вільна група $T(\Omega, X, Eq)$ називається* **вільною абелевою групою**, *якщо, крім тотожних співвідношень, які визначають її як вільну групу, множина Eq включає тотожне співвідношення комутативності.*

Довільне слово із $T(\Omega, X, Eq)$ можна записати у вигляді

$$x_1^{n_1} x_2^{n_2} \ldots x_k^{n_k}, \tag{14}$$

де $x_i \in X$, x_i всі попарно різні, n_i — цілі числа, $j = 1, 2, \ldots, k$. Запис слова у вигляді (14) називають *мультиплікативним*, але часто для запису слів абелевої групи (як і слів комутативної напівгрупи) використовують адитивний запис — у вигляді суми:

$$n_1 \cdot x_1 + n_2 \cdot x_2 + \ldots + n_k \cdot x_k. \tag{15}$$

Числа n_i у цьому виразі називаються **коефіцієнтами.** Додавання слів вигляду (15) визначається як додавання коефіцієнтів за однакових елементів $x_i \in X$, а роль одиниці у такій формі запису відіграє **нуль** — нульовий елемент, тобто вираз вигляду (15), у якого всі коефіцієнти дорівнюють нулю. Якщо деякі x_i із X у виразі (15) відсутні, то вважається, що коефіцієнт n_i при цьому x_i дорівнює нулю.

Приклади абелевих груп

1. Множина $M = \{-1, 1\}$ являє собою групу відносно звичайного множення чисел. Дійсно, $1 \cdot 1 = 1$, $(-1) \cdot 1 = 1 \cdot (-1) = -1$, $(-1) \cdot (-1) = 1$. Отже, множина M замкнута відносно операції множення. Роль одиниці виконує елемент 1. Ця множина також замкнута і відносно операції взяття оберненого (елементи 1 і –1 обернені до самих себе). Очевидно, що операція множення асоціативна і комутативна.

2. Множина $M = \{0, 1\}$ складає абелеву групу відносно операції додавання, якщо припустити $1 + 1 = 0$ (група лишків за модулем 2). Дійсно, $0 + 0 = 0$, $0 + 1 = 1 + 0 = 1$, $1 + 1 = 0$ і, отже, M замкнута відносно операції додавання і взяття оберненого (1 є елементом оберненим до самого себе). Одиничним елементом служить 0. Очевидно, що операція додавання асоціативно-комутативна.

Дану конструкцію можна узагальнити. Нехай $Z_n = \{0, 1, \ldots, n-1\}$, $n > 1$. Задамо на множині Z_n операцію додавання $\oplus$:

$$k \oplus l = \begin{cases} k + l, & \text{якщо } k + l < n, \\ k + l - n, & \text{якщо } k + l \geq n, \end{cases}$$

де + і – звичайні операції додавання і віднімання цілих чисел, k, $l \in Z_n$. Множина Z_n із заданою таким чином операцією додавання складає абелеву групу n-го порядку. Дійсно, роль нуля відіграє елемент 0, роль оберненого елемента до даного елемента k — елемент l такий, що $k + l = n$. Очевидно, що операція додавання комутативна в силу комутативності операції додавання цілих чисел. Доведення асоціативності операції пропонується як проста вправа.

Група Z_n називається **групою лишків за модулем** n і виникає в результаті розбиття множини цілих чисел Z на класи, де в один клас потрапляють ті і тільки ті числа, які при діленні на $n > 1$ дають однакові остачі. Якщо позначити $rest(m, n)$ остачу від ділення числа m на n, то операцію додавання $\oplus$ можна визначити іншим способом:

$$k \oplus l = rest(k + l, n).$$

Очевидно, що розглянута вище група $M = \{0, 1\}$ збігається з групою Z_2.

3. Множина цілих чисел Z складає адитивну групу, але Z не є групою відносно операції множення, оскільки операція ділення в множині Z (операція взяття оберненого) не завжди визначена.

Множина Z_0 цілих парних чисел складає адитивну групу і є підгрупою групи Z. Загалом адитивною підгрупою буде довільна множина цілих чисел, які кратні деякому заданому цілому числу n.

Множина непарних чисел не буде групою, оскільки вона не замкнута відносно операції додавання.

4. Множина раціональних чисел RC відносно додавання являє собою групу, але відносно множення RC не буде групою, оскільки ділення на 0 неможливе. Зауважимо, що $RC \setminus \{0\}$ — мультиплікативна група раціональних чисел.

5. Циклічна група G, тобто група, породжена одним елементом, наприклад a, складається із елементів вигляду a^n, де $n \in Z$, $a^0 = e$, $a^1 = a$. Множення елементів визначається як додавання степенів, тобто $a^k \cdot a^l = a^{k+l}$. ♠

Приклад гомоморфізму груп

Адитивна група цілих чисел Z гомоморфно відображається на мультиплікативну групу M (див. приклади абелевих груп, приклад 1)). Відображення h задається таким чином:

$$h(n) = \begin{cases} 1, & \text{якщо } n \text{ — парне,} \\ -1, & \text{якщо } n \text{ — непарне.} \end{cases}$$

Покажемо, що h — гомоморфізм. Спочатку покажемо, що

$$h(m + n) = h(m) \cdot h(n).$$

Можливі такі чотири випадки:

а) $m = 2k$, $n = 2l$; *b*) $m = 2k$, $n = 2l + 1$;
c) $m = 2k + 1$, $n = 2l$; *d*) $m = 2k + 1$, $n = 2l + 1$.

У випадку а) маємо $h(m + n) = h(2k + 2l) = h(2(k + l) = 1$. З іншого боку, $h(m)h(n) = h(2k)h(2l) = 1 \cdot 1 = 1$.

У випадку b) ((c)) маємо $h(m+n) = h(2(k+l)+1) = -1$. З іншого боку, $h(m) \cdot h(n) = h(2k) \cdot h(2l+1) = 1 \cdot (-1) = -1$ ($h(m) \cdot h(n) = h(2k+1) \cdot h(2l) = (-1) \cdot 1 = -1$).

І у випадку d) маємо $h(m+n) = h(2k+1) \cdot h(2l+1) = h(2(k+l+1)) = 1$. З іншого боку, $h(m) \cdot h(n) = h(2k+1) \cdot h(2l+1) = (-1) \cdot (-1) = 1$.

Отже, рівність $h(m+n) = h(m) \cdot h(n)$ виконується. Далі, зважаючи на те, що $h(0) = 1$, нульарні операції теж задовольняють цій рівності. І останнє, якщо

$$h(n) = \begin{cases} 1, & \text{якщо } n \text{ — парне,} \\ -1, & \text{якщо } n \text{ — непарне,} \end{cases}$$

тоді

$$h(-n) = \begin{cases} 1, & \text{якщо } n \text{ — парне,} \\ -1, & \text{якщо } n \text{ — непарне.} \end{cases}$$

Оскільки елементи 1 і –1 обернені самі до себе, то і в цьому випадку все коректно, так як обернені елементи переходять у відповідні обернені.

Очевидно, що це відображення не взаємно однозначне. Таким чином, побудоване відображення h — гомоморфізм. ♠

Приклад ізоморфізму груп

Теорема 41. *Довільна нескінченна циклічна група ізоморфна адитивній групі цілих чисел Z. Довільна скінченна циклічна група n-го порядку ізоморфна групі лишків за модулем n – Z_n.*

Доведення. У першому випадку відображення h задається $h(a^n) = n$, а в другому $h(a^k) = rest(k, n)$.

Покажемо, що h — ізоморфізм для першого випадку.

a1) $h(a^m \cdot a^n) = h(a^{m+n}) = m + n$;

a2) $h(a^{-n}) = -n$,

і оскільки a^n обернений до a^{-n}, то умови ізоморфізму виконуються;

a3) $h(a^0) = h(e) = 0$.

Залишається показати, що h — взаємно однозначне. Для цього необхідно зауважити, що із $a^m = a^n$ випливає $a^{m+n} = a^0 = e$, тобто $m = n$. ■

Другий випадок пропонується розглянути як вправу. ♠

Приклад побудови «нетрадиційної» арифметики на основі абелевих груп

Нехай задана деяка скінченна множина цілих чисел, наприклад, $N_5 = \{0, 1, 2, 3, 4\}$. Оскільки ми хочемо побудувати адитивну абелеву групу, то ця множина обов'язково повинна включати 0. Для того

щоб N_5 перетворити в групу GN_5, необхідно коректно задати значення для операції додавання з одним із елементів групи, скажімо з 1. Дійсно, оскільки $a + 0 = a$ для довільного $a \in GN_5$, то перший рядок таблиці додавання елементів групи визначений (табл. 1), а в силу симетричності (оскільки GN_5 абелева) і перший стовпчик цієї таблиці. Нехай, наприклад, задано $0 + 1 = 1$, $1 + 1 = 4$, $1 + 4 = 2$, $1 + 2 = 3$, $1 + 3 = 0$. Таке задання коректне, оскільки має місце єдиність результату (але єдиність результату, як буде показано нижче, не достатня умова гарантії коректності). Тепер послідовно знаходимо (табл. 2):

$2 + 2 = 2 + (1 + 4) = (2 + 1) + 4 = 3 + (1 + 1) = (3 + 1) + 1 = 0 + 1 = 1,$

$2 + 3 = 2 + (1 + 2) = (2 + 2) + 1 = 1 + 1 = 4,$

$2 + 4 = 2 + (1 + 1) = (2 + 1) + 1 = 3 + 1 = 0,$

$3 + 3 = 3 + (1 + 2) = (3 + 1) + 2 = 0 + 2 = 2,$

$3 + 4 = 3 + (1 + 1) = (3 + 1) + 1 = 0 + 1 = 1,$

$4 + 4 = 4 + (1 + 1) = (4 + 1) + 1 = 2 + 1 = 3.$

Заносимо ці значення в таблицю 2 і на цьому закінчуємо побудову групи GN_5.

Таблиця 1

+	0	1	2	3	4
0	0	1	2	3	4
1	1	4	3	0	2
2	2	3			
3	3	0			
4	4	2			

Таблиця 2

+	0	1	2	3	4
0	0	1	2	3	4
1	1	4	3	0	2
2	2	3	1	4	0
3	3	0	4	2	1
4	4	2	0	1	3

Аналогічно можна задати й довільну іншу групу GN_5, яка включає 0. Дійсно, якщо, наприклад, $M_5 = \{0, 2, 6, 3, 5\}$, то встановимо взаємно однозначну відповідність h між N_5 і M_5 (ця відповідність існує, оскільки N_5 і M_5 рівнопотужні), яка переводить 0 у 0, а останні елементи — довільним чином. Наприклад, нехай задано відповідність

$$1 \leftrightarrow 6,\ 2 \leftrightarrow 3,\ 3 \leftrightarrow 5,\ 4 \leftrightarrow 2,\ 0 \leftrightarrow 0.$$

Звідси отримуємо таку таблицю.

Таблиця 3

+	0	2	6	3	5
0	0	2	6	3	5
2	2	5	3	0	6
6	6	3	2	5	0
3	3	0	5	6	2
5	5	6	0	2	3

Перевіримо, наприклад, чому відповідає в цій таблиці $4 = 1 + 1$. Маємо $h(1 + 1) = h(4) = h(1) + h(1) = 6 + 6 = 2$.

Як вправа, пропонується виконання перевірки всієї таблиці 3.

Зауважимо, що для побудови групи GN_k, мало вимагати тільки однозначності операції додавання. Якщо визначити додавання в групі так, що в ній буде елемент скінченного порядку (тобто група з крученням), то коректність визначення операції може бути порушена. Наприклад, якщо задати додавання

$$0 + 1 = 1,\ 1 + 1 = 0,\ 1 + 2 = 3,\ 1 + 3 = 4,\ 1 + 4 = 2,$$

то, виконуючи перевірку (обчислюючи) $1 + 3$, отримаємо

$$1 + 3 = 1 + (1 + 2) = (1 + 1) + 2 = 0 + 2 = 2,$$

що не збігається з визначеним вище. Справа в тім, що елемент 1 є елементом 2-го порядку ($1 + 1 = 0$), і це вносить свої корективи при визначенні операції (її вже не можна задавати довільним чином). ♠

2.2.7. Вільні кільця

Визначення 43. *$T(\Omega, X, Eq)$ називається* **вільним кільцем**, *якщо Eq визначає $T(\Omega, X, Eq)$ як*

1) вільну абелеву групу відносно додавання;

2) вільний групоїд відносно множення;

3) і включає закони дистрибутивності, тобто для довільних x, x', x'' із $T(\Omega, X, Eq)$

$$x(x' + x'') = (xx') + (xx''),$$
$$(x + x')x'' = (xx'') + (x'x'').$$

Це означає, що Ω включає чотири операції — бінарні операції додавання і множення, унарну операцію взяття оберненого відносно операції додавання і нульарну операцію, яка фіксує нульовий елемент абелевої групи кільця. Цей нульовий елемент називається **нульовим** елементом кільця.

Множення у вільному кільці зводиться, на основі законів дистрибутивності, до множення елементів із X, яке виконується за правилом множення слів у вільному групоїді.

Із законів 1) — 3) випливають співвідношення, які дає

Твердження 7. *У довільному кільці K для довільних його елементів a, b, c має місце:*

a) $a(b - c) = ab - ac,\ \ (b - c)a = ba - ca;$

b) $a0 = 0a = 0;$

c) $(-a)b = a(-b) = -ab,\ \ (-a)(-b) = ab.$

Доведення. а) На основі комутативності та асоціативності операції додавання можна записати $c + (b - c) = b$. Домноживши обидві частини цього рівняння зліва на a, маємо

$$a(c + (b - c)) = ac + a(b - c) = ab.$$

Звідси, $(ac + a(b - c)) - ac = ab - ac = a(b - c)$, знову ж таки на основі комутативності та асоціативності операції додавання.

Аналогічно доводиться і другий закон.

b) Дійсно, нехай b — довільний елемент кільця. Тоді, згідно із законом а), доведеним вище, маємо

$$a0 = a(b - b) = ab - ab = 0.$$

c) Доведення пропонується як вправа.

Зауважимо, що обернене твердження до твердження *b)* у довільному кільці не має місця, тобто існують такі кільця, в яких є відмінні від нуля елементи a, b, добуток яких дорівнює нулю ($ab = 0$). Якщо такі елементи в кільці є, то вони називаються **дільниками нуля.**

Визначення 44. *Кільце називається* **асоціативним (комутативним),** *якщо операція множення асоціативна (комутативна), і називається* **кільцем з одиницею,** *якщо воно має одиничний елемент відносно операції множення.*
Кільце називається **асоціативно-комутативним,** *якщо воно асоціативне і комутативне одночасно.*

Елементами вільного асоціативно-комутативного кільця з множиною вільних твірних X є многочлени від елементів із X з цілими коефіцієнтами. Тому таке кільце часто називають просто **кільцем многочленів над** X.

Множення у вільному асоціативному кільці виконується за правилом множення слів у вільній напівгрупі, а сама напівгрупа називається **мультиплікативною напівгрупою** асоціативного кільця.

Приклади кілець

1. Прикладом асоціативного кільця з одиницею може служити множина квадратних матриць над довільним кільцем P з одиницею.

Нехай $M(p, q, P)$ — множина всіх матриць розмірності $p \times q$ над кільцем P, а $M(p, P)$ — множина квадратних матриць $M(p, p, P)$.

Матриця C, яка складається з елементів вигляду $a_{ij} + b_{ij}$, де a_{ij}, b_{ij} — відповідно елементи матриць A B із $M(p, q, P)$, називається **сумою матриць** A і B. Оскільки множина P є кільцем, то множина $M(p, q, P)$, очевидно, є абелевою групою відносно додавання. Роль нульового елемента відіграє матриця, у якої всі елементи склада-

ються із нулів (*нульова матриця*), а роль оберненої матриці для матриці A відіграє матриця A', у якої $a'_{ij} = -a_{ij}$, де a'_{ij}, a_{ij} — елементи матриць A' і A відповідно.

Нехай $A \in M(p, q, P)$, $B \in M(q, r, P)$, тоді матриця C, яка складається з елементів

$$c_{ij} = \sum_{k=1}^{q} a_{ik} \cdot b_{kj}$$

називається **добутком матриць** A і B. Із цього визначення випливає, що не довільні дві матриці можна перемножити, а лише *відповідні*, тобто такі, у яких число стовпчиків першої дорівнює числу рядків другої. Зрозуміло, що квадратні матриці із $M(p, P)$ відповідні, і для таких матриць добуток завжди визначений.

Розглянемо множину квадратних матриць $M(p, P)$. Матриця A називається **одиничною,** якщо $a_{ij} = 0$ при $i \neq j$ і $a_{ii} = 1_P$, де 1_P — одиниця кільця P. Одинична матриця, як правило, позначається через E. Легко впевнитися, що $\forall A \in M(p, P)$ має місце рівність $A \cdot E = E \cdot A = A$, тобто матриця E відіграє роль одиничного елемента у множині $M(p, P)$.

Неважко показати, що $M(p, P)$ є асоціативним кільцем з одиницею. Покажемо, наприклад, справедливість одного із дистрибутивних законів:

$$(\forall A, B \in M(p, P))\ A \cdot (B + C) = A \cdot B + A \cdot C.$$

Дійсно, за визначенням добутку матриць маємо

$$\sum_{k=1}^{p} a_{ik} \left(b_{kj} + c_{kj}\right) = \sum_{k=1}^{p} \left(a_{ik} \cdot b_{kj} + a_{ik} \cdot c_{kj}\right) = \sum_{k=1}^{p} a_{ik} \cdot b_{kj} + \sum_{k=1}^{p} a_{ik} \cdot c_{kj},$$

тобто $A \cdot (B + C)$ і $A \cdot B + A \cdot C$ мають однакові елементи.

2. Множина $M = \{0, 1\}$ буде кільцем, якщо визначити додавання і множення елементів цієї множини так:

$$0 + 0 = 1 + 1 = 0,\ 0 + 1 = 1 + 0 = 1,\ 0 \cdot 1 = 1 \cdot 0 = 0,\ 1 \cdot 1 = 1.$$

Неважко перевірити, що ця множина буде асоціативно-комутативним кільцем з одиницею.

Користуючись квадратною матрицею над цим кільцем, яка відповідає деякому бінарному відношенню R на скінченній множині A (див. розділ 1), можна обчислити транзитивне замикання цього відношення R_t.

Розглянемо приклад. Нехай множина $A = \{a_1, a_2, a_3, a_4\}$, а відношення $R = \{(a_1, a_1), (a_1, a_2), (a_2, a_3), (a_2, a_4), (a_3, a_4), (a_4, a_1), (a_4, a_3)\}$, тоді матриця $A(R)$ має вигляд:

$$A(R)=\begin{pmatrix}1 & 1 & 0 & 0\\ 0 & 0 & 1 & 1\\ 0 & 0 & 0 & 1\\ 1 & 0 & 1 & 0\end{pmatrix}.$$

Для вправи пропонується показати, що $A(R)^2$ відповідає відношенню R^2, $A(R)^3$ — відношенню R^3 і т. д.

Загалом, якщо R і R_1 — деякі бінарні відношення на множині A, яким відповідають матриці $A(R)$ і $A(R_1)$, то відношенню $R * R_1$ відповідає матриця яка є добутком матриць $A(R)$ і $A(R_1)$ (див. вправу 13).

3. Розглядаючи абелеві групи, ми будували приклад «нетрадиційної арифметики» для операції додавання. Кільця дозволяють розширити таку побудову і на операцію множення.

Розглянемо спочатку, як довизначається група GN_5 (див. табл. 2) до кільця з одиницею. Роль одиниці буде відігравати 1. Відповідно до аксіом кільця і твердження 7 маємо: для довільного елемента a із GN_5 $a \cdot 1 = 1 \cdot a = a$, $0 \cdot a = 0 \cdot a = 0$. Таким чином, два рядки і два стовпчики таблиці множення вже визначені. Далі, використовуючи таблицю 2 для додавання і закон дистрибутивності, одержуємо

Таблиця 4

·	0	1	2	3	4
0	0	0	0	0	0
1	0	1	2	3	4
2	0	2	3	4	1
3	0	3	4	1	2
4	0	4	1	2	3

оскільки

$$4 \cdot 2 = (1 + 1) \cdot 2 = 2 + 2 = 1;$$
$$4 \cdot 3 = (1 + 1) \cdot 3 = 3 + 3 = 2;$$
$$4 \cdot 4 = (1 + 1) \cdot 4 = 4 + 4 = 3$$

Легко пересвідчитися, що $4 \cdot 2 = 2 \cdot 4$ і $4 \cdot 3 = 3 \cdot 4$, тобто елемент 4 комутативний зі всіма останніми. Далі

$$2 \cdot 2 = (1 + 4) \cdot 2 = 2 + 4 \cdot 2 = 2 + 1 = 3;$$
$$2 \cdot 3 = (1 + 4) \cdot 3 = 3 + 4 \cdot 3 = 3 + 2 = 4;$$
$$2 \cdot 4 = (1 + 4) \cdot 4 = 4 + 4 \cdot 4 = 4 + 3 = 1.$$

Аналогічно одержуємо $3 \cdot 3 = (2 + 1) \cdot 3 = 2 \cdot 3 + 3 = 4 + 3 = 1$ і всі останні елементи таблиці 4. Із симетричності таблиці випливає, що множина елементів GN_5 комутативна. Крім того, легко перевірити,

що GN_5 також асоціативна, тобто GN_5 — асоціативно-комутативне кільце з одиницею.

Розглянемо тепер приклад кільця без одиниці. Візьмемо групу GM5, операція додавання якої задана таблицею 3. Для того щоб перетворити цю групу на кільце, необхідно, як і у випадку додавання, задати хоча б один рядок для операції множення.

Нехай задано: $2 \cdot 2 = 3$, $2 \cdot 3 = 5$, $2 \cdot 5 = 6$, $2 \cdot 6 = 2$. Тоді маємо:

$5 \cdot 2 = (2 + 2) \cdot 2 = 2 \cdot 2 + 2 \cdot 2 = 3 + 3 = 6;$
$5 \cdot 3 = (2 + 2) \cdot 3 = 2 \cdot 3 + 2 \cdot 3 = 5 + 5 = 3;$
$5 \cdot 6 = (2 + 2) \cdot 6 = 2 \cdot 6 + 2 \cdot 6 = 2 + 2 = 5;$
$5 \cdot 5 = (2 + 2) \cdot 5 = 2 \cdot 5 + 2 \cdot 5 = 6 + 6 = 2;$
$6 \cdot 2 = (5 + 2) \cdot 2 = 5 \cdot 2 + 2 \cdot 2 = 6 + 3 = 5;$
$6 \cdot 3 = (5 + 2) \cdot 3 = 5 \cdot 3 + 2 \cdot 3 = 3 + 5 = 2;$
$6 \cdot 5 = (5 + 2) \cdot 5 = 5 \cdot 5 + 2 \cdot 5 = 2 + 6 = 3;$
$6 \cdot 6 = (5 + 2) \cdot 6 = 5 \cdot 6 + 2 \cdot 6 = 5 + 2 = 6;$
$3 \cdot 2 = (6 + 2) \cdot 2 = 6 \cdot 2 + 2 \cdot 2 = 5 + 3 = 2;$
$3 \cdot 3 = (6 + 2) \cdot 3 = 6 \cdot 3 + 2 \cdot 3 = 2 + 5 = 6;$
$3 \cdot 6 = (6 + 2) \cdot 6 = 6 \cdot 6 + 2 \cdot 6 = 6 + 2 = 3;$
$3 \cdot 5 = (6 + 2) \cdot 5 = 6 \cdot 5 + 2 \cdot 5 = 3 + 6 = 5.$

тобто, одержуємо таку таблицю множення:

Таблиця 5

·	0	2	6	3	5
0	0	0	0	0	0
2	0	3	2	5	6
6	0	5	6	2	3
3	0	2	3	6	5
5	0	6	5	3	2

Із цієї таблиці видно, що кільце некомутативне. Легко впевнитися, що воно і неасоціативне, наприклад $(3 \cdot 6) \cdot 5 = 3 \cdot 5 = 5$ і $3 \cdot (6 \cdot 5) = 3 \cdot 3 = 6$ не рівні між собою.

Зауважимо, що для знаходження оптимальним шляхом таблиці як для додавання, так і для множення найкраще це робити таким чином. Якщо визначено множення на число 2 (як у попередньому прикладі), то в таблиці додавання знаходимо суму 2 + 2 (вона дорівнює 5) і беремо результат цієї суми як перший множник. Далі чинимо аналогічно для числа 2 і числа, яке отримане раніше (у прикладі це число 5), шукаємо добуток цих чисел. Знаходимо результат множення для них і т. д., доки не побудуємо всі елементи таблиці.

При обчисленнях на ЕОМ така методика часто відіграє дуже важливу роль для побудови ефективних алгоритмів.

4. Асоціативно-комутативне кільце називається **булевим**, якщо для довільного елемента a із цього кільця має місце співвідношення $a^2 = a$. Ця тотожність називається **законом ідемпотентності.**

Твердження 8. *У довільному булевому кільці для довільних елементів a, b мають місце співвідношення*

$$\text{(a) } 2 \cdot a = 0; \quad \text{(b) } a \cdot b = b \cdot a.$$

Доведення. (a) В силу ідемпотентності $a + b = (a + b)^2 = a + a \cdot b + b \cdot a + b$. Звідси отримуємо $a \cdot b + b \cdot a = 0$. Покладемо $b = a$, тоді $a^2 + a^2 = a + a = 2 \cdot a = 0$. Звідси знаходимо також, що $a = -a$.

(b) Із (a) одержуємо $a \cdot b = b \cdot a = 0$. Змінюючи a на $-a$ в одному із доданків, знаходимо $a \cdot b - b \cdot a = 0$ або $a \cdot b = b \cdot a$.

Що й потрібно було довести ♠

Асоціативно-комутативне кільце без дільників нуля називається **областю цілісності.**

Областями цілісності, як легко переконатися, будуть кільця цілих і раціональних чисел — Z і RC. Областю цілісності буде також і вільне кільце многочленів.

Останній приклад з многочленами підказує конструкцію, за допомогою якої можна будувати інші приклади областей цілісності.

Нехай KC — довільне асоціативно-комутативне кільце. Розглянемо всі можливі многочлени вигляду

$$a_0 + a_1 \cdot x_1 + a_2 \cdot x_2 + \ldots + a_n \cdot x_n,$$

де $n_i \in N$, відносно невідомого x з коефіцієнтами a_i із KC, $i = 1,\ldots, n$. Якщо $a_n \neq 0$, то число n називається *степенем* цього многочлена. Визначаючи додавання і множення многочленів звичайним шляхом, так як це прийнято в курсі вищої алгебри, отримуємо **кільце многочленів** $R[x]$ від параметра x над кільцем KC. Нулем кільця $R[x]$ служить многочлен, усі коефіцієнти якого дорівнюють нулю.

Аналогічно можна визначити кільце многочленів $R[x_1, x_2,\ldots, x_n]$ від довільного скінченного числа невідомих як кільце многочленів від одного невідомого x_n над кільцем $R[x_1, x_2, \ldots, x_{n-1}]$. Має місце така теорема.

Теорема 42. *Якщо KC — область цілісності, то* $R[x_1, x_2, \ldots, x_n]$ *теж область цілісності* (*див. вправу* 17).

Визначення 45. *Асоціативно-комутативне кільце з одиницею називається* **полем,** *якщо воно як відносно додавання, так і відносно множення є абелевою групою. Група поля відносно додавання називається* адитивною, *а група відносно множення* — мультиплікативною.

Зауважимо, що поле є областю цілісності і через це, як випливає із теореми 42, кільце многочленів над довільним полем P є областю цілісності. Одиницею в цій області цілісності служить многочлен, коефіцієнти якого всі дорівнюють нулю, крім коефіцієнта a_0, який дорівнює 1_P — одиниці поля P.

2.2.8. Векторні простори

Визначення 46. *$T(\Omega, X, Eq)$ називається* **векторним простором над деяким полем** *P, якщо Ω включає бінарну операцію додавання елементів вигляду* (15), *нескінченне число унарних операцій множення елементів вигляду* (15) *на елементи із поля P (для кожного a із P своя операція) і нульарну операцію, яка фіксує нульовий елемент* — 0, *а множина Eq включає всі співвідношення, що визначають $T(\Omega, X, Eq)$ як вільну абелеву групу, тобто для довільних x, x', x'' із $T(\Omega, X, Eq)$*

$$x + (x' + x'') = (x + x') + x'';$$
$$x + x' = x' + x;$$
$$x + 0 = x;$$
$$x + (-x) = 0.$$

а також для довільних x, x' із $T(\Omega, X, Eq)$ і a, b із P мають місце рівності

$$a \cdot (x + x') = a \cdot x + a \cdot x';$$
$$(a + b) \cdot x = a \cdot x + b \cdot x;$$
$$a \cdot (b \cdot x) = (a \cdot b) \cdot x.$$

Елементами $T(\Omega, X, Eq)$ є елементи вигляду (15), які називаються **векторами.** Символам x_i із X співставляється вектор

$$0 \cdot x_1 + 0 \cdot x_2 + \ldots + 1 \cdot x_i + \ldots + 0 \cdot x_n + \ldots, \qquad (16)$$

а роль нульового вектора простору відіграє елемент

$$0 \cdot x_1 + 0 \cdot x_2 + \ldots + 0 \cdot x_i + \ldots + 0 \cdot x_n + \ldots = 0,$$

де $i = 1, 2, \ldots$.

Нехай $v_1, v_2, \ldots, v_n \in T(\Omega, X, Eq)$ і $m_1, m_2, \ldots, m_n \in P$, тоді суму вигляду

$$m_1 \cdot v_1 + m_2 \cdot v_2 + \ldots + m_n \cdot v_n = \sum_{i=1}^{n} m_i \cdot v_i$$

називають **лінійною комбінацією векторів** $v_1, v_2,..., v_n$. Говорять, що вектори $v_1, v_2,..., v_n$ **лінійно незалежні**, якщо

$$\sum_{i=1}^{n} m_i \cdot v_i = 0 \Leftrightarrow m_1 = m_2 = ... = m_n = 0,$$

у протилежному випадку вектори $v_1, v_2,..., v_n$ називаються **лінійно залежними.**

Припустимо, що в деякому просторі L над полем P існує лінійно незалежна система векторів $v_1, v_2,..., v_n$ і немає ніякої лінійно незалежної системи, яка складається з більшого ніж n числа векторів. Тоді говорять, що L — ***n*-мірний векторний простір** над полем P. Число n називають розмірністю простору L, а довільну сукупність лінійно незалежних векторів $v_1, v_2,..., v_n$ із L — **базисом *L*.** Якщо такого числа n немає, то простір L називається **нескінченномірним.**

Зауважимо, що вектори вигляду (16) лінійно незалежні, і тому X можна розглядати як базис $T(\Omega, X, Eq)$, і якщо X — скінченна множина, то $T(\Omega, X, Eq)$ — скінченномірний векторний простір над P.

Теорема 43. *Якщо X — базис n-мірного векторного простору $T(\Omega, X, Eq)$ над полем P, то для довільного вектора $u \in T(\Omega, X, Eq)$ існує єдина можливість запису його у вигляді лінійної комбінації*

$$u = m_1 \cdot x_1 + m_2 \cdot x_2 + ... + m_n \cdot x_n$$

базисних векторів $x_1, x_2,..., x_n \in X$.

Доведення. Оскільки X — базис, то сукупність векторів $u, x_1, x_2, ..., x_n$ в n-мірному векторному просторі $T(\Omega, X, Eq)$ лінійно залежна, у той час як $x_1, x_2, ..., x_n$ — незалежна система. Отже, $u = m_1 \cdot x_1 + m_2 \cdot x_2 + ... + m_n \cdot x_n$.

Якби існувала інша можливість запису вектора u

$$u = l_1 \cdot x_1 + l_2 \cdot x_2 + ... + l_n \cdot x_n,$$

то

$$u - u = (m_1 - l_1) \cdot x_1 + (m_2 - l_2) \cdot x_2 + ... + (m_n - l_n) \cdot x_n = 0.$$

Звідси, оскільки вектори $x_1, x_2,..., x_n$ лінійно незалежні, одержуємо що $m_1 = l_1, m_2 = l_2,..., m_n = l_n$. ■

Однозначно визначені коефіцієнти $m_1, m_2,..., m_n$ вектора u із $T(\Omega, X, Eq)$ називаються його **координатами** в базисі $v_1,..., v_n$.

Якщо $L \subseteq T(\Omega, X, Eq)$, то множина L називається **підпростором простору** $T(\Omega, X, Eq)$, якщо вона разом з векторами $v_1, v_2,..., v_k$ включає і довільну їх лінійну комбінацію $m_1 \cdot v_1 + m_2 \cdot v_2 + ... + m_k \cdot v_k$. В окремому випадку, якщо $T(\Omega, X, Eq)$ — n-мірний векторний прос-

тір і L його підпростір, то зрозуміло, що розмірність L не перевищує розмірності всього простору $T(\Omega, X, Eq)$. Має місце такий факт.

Теорема 44. *Якщо підпростір L n-мірного векторного простору $T(\Omega, X, Eq)$ має ту саму розмірність, що й $T(\Omega, X, Eq)$, то він збігається з усім простором $T(\Omega, X, Eq)$.*

Доведення. Нехай L і $T(\Omega, X, Eq)$ мають одну й ту саму розмірність n. Візьмемо базис простору L — $v_1, v_2, ..., v_n$. Ця система буде базисом обох просторів (оскільки $v_1, v_2, ..., v_n \in T(\Omega, X, Eq)$). Отже, для довільного $u \in T(\Omega, X, Eq)$ має місце розклад

$$u = m_1 \cdot v_1 + m_2 \cdot v_2 + ... + m_n \cdot v_n$$

і $u \in L$, оскільки u — лінійна комбінація векторів із L, тобто $T(\Omega, X, Eq) \subseteq L$, а отже,

$$L = T(\Omega, X, Eq). \blacksquare$$

2.3. Булеві алгебри

Визначення 47. *Алгебра $G = (A, \Omega) \in (\Omega, Eq)$ називається* **булевою алгеброю,** *якщо Ω складається із*

б1) *двох бінарних операцій — $\vee$ (або) і $\wedge$ (і);*

б2) *однієї унарної операції — $\neg$ (заперечення);*

б3) *двох нульарних операцій — 0 (нуль) і 1 (одиниця), і для довільних $a, b, c \in A$ має місце така сукупність співвідношень Eq:*

b1) $a \vee b = b \vee a$, $a \wedge b = b \wedge a$ — *комутативність;*

b2) $a \vee (b \vee c) = (a \vee b) \vee c$, $a \wedge (b \wedge c) = (a \wedge b) \wedge c$ — *асоціативність;*

b3) $a \vee (b \wedge c) = (a \vee b) \wedge (a \vee c)$,
$a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c)$ — *дистрибутивність;*

b4) $a \vee 0 = a$, $a \vee \neg a = 1$, $a \wedge 1 = a$, $a \wedge \neg a = 0$ — *закони для нуля, одиниці і заперечення.*

Як наслідки співвідношень b1) — b4), можна одержати деякі корисні співвідношення. Наведемо основні з них.

Наслідок 15 (*Закон ідемпотентності*). *Для довільного елемента a із A мають місце рівності*

$$a \vee a = a \wedge a = a.$$

Доведення. Використовуючи закони b4) і закони дистрибутивності, маємо

$$a \vee a = (a \vee a) \wedge 1 = (a \vee a) \wedge (a \vee \neg a) = a \vee (a \wedge \neg a) = a \vee 0 = a,$$
$$a \wedge a = (a \wedge a) \vee 0 = (a \wedge a) \vee (a \wedge \neg a) = a \wedge (a \vee \neg a) = a \wedge 1 = a.$$

Звідси як наслідок одержуємо

$$a \vee 1 = a \vee (a \vee \neg a) = (a \vee a) \vee \neg a = a \vee \neg a = 1;$$
$$a \wedge 0 = a \wedge (a \wedge \neg a) = (a \wedge a) \wedge \neg a = a \wedge \neg a = 0.$$

Наслідок 16 (*Закон поглинання*). *Для довільних елементів* $a, b \in A$ *мають місце рівності* $a \wedge (a \vee b) = a \vee (a \wedge b) = a$.

Доведення. Доведемо рівність $a \vee (a \wedge b) = a$. Використовуючи співвідношення b4), закон дистрибутивності і наслідок 15, маємо

$$a \vee (a \wedge b) = (a \wedge 1) \vee (a \wedge b) = a \wedge (1 \vee b) = a \wedge 1 = a.$$

Аналогічно доводиться і друга рівність:

$$a \wedge (a \vee b) = (a \vee 0) \wedge (a \vee b) = a \vee (0 \wedge b) = a \vee 0 = a.$$

Наслідок 17. *Для довільних елементів* $a, b, c \in A$ *із рівностей*

$$a \wedge c = b \wedge c \text{ і } a \vee c = b \vee c$$

випливає рівність $a = b$.

Доведення. За законом поглинання маємо $a = a \vee (a \wedge c)$. Застосовуючи закони дистрибутивності і рівності, наведені в умові, отримуємо

$$a = a \vee (a \wedge c) = a \vee (b \wedge c) = (a \vee b) \wedge (a \vee c) = (a \vee b) \wedge (b \vee c) =$$
$$b \vee (a \wedge c) = b \vee (b \wedge c) = b,$$

що й було потрібно показати.

Наслідок 18 (*Закон подвійного заперечення*). *Для довільного елемента* a *із* A *мають місце рівності*

$$\neg(\neg a) = a, \ \neg(0) = 1, \ \neg(1) = 0.$$

Дійсно, $\neg(a) \vee \neg(\neg(a)) = 1$ і $\neg(a) \wedge \neg(\neg(a)) = 0$, $\neg(a) \vee a = 1$ і $\neg(a) \wedge a = 0$ згідно з b4). Звідси випливає, що $\neg(a) \vee \neg(\neg(a)) = \neg(a) \vee a$ і $\neg(a) \wedge \neg(\neg(a)) = \neg(a) \wedge a$. За наслідком 17 отримуємо $a = \neg(\neg(a))$.

Аналогічно із $\neg 0 \vee 0 = 1 \vee 0 = 1$ і $\neg 0 \wedge 0 = 1 \wedge 0 = 0$ випливає $\neg 0 = 1$.

Наслідок 19 (*Закони де Моргана*). *Для довільних елементів* $a, b \in A$ *мають місце рівності*

$$\neg(a \vee b) = \neg a \wedge \neg b, \neg(a \wedge b) = \neg a \vee \neg b.$$

Доведення. Розглянемо вираз

$$(a \vee b) \wedge (\neg a \wedge \neg b) = (a \wedge (\neg a \wedge \neg b)) \vee (b \wedge (\neg a \wedge \neg b)) = 0 \vee 0 = 0.$$

Далі

$$(a \vee b) \vee (\neg a \wedge \neg \mathrm{b}) = ((a \vee b) \vee \neg(a)) \wedge ((a \vee b) \vee \neg b) = 1 \wedge 1 = 1.$$

Згідно з b4) маємо

$$(a \vee b) \wedge (\neg a \wedge \neg \mathrm{b}) = (a \vee b) \wedge \neg(a \vee b) = 0$$

і

$$(a \vee b) \vee (\neg a \wedge \neg b) = (a \vee b) \vee \neg(a \vee b) = 1.$$

Звідси за наслідком 17 маємо, що $\neg(a \vee b) = \neg a \wedge \neg b$.

Аналогічно доводиться і другий закон. ■

Операції булевої алгебри $\vee$, $\wedge$, $\neg$ відповідно називають **диз'юнкцією, кон'юнкцією** і **запереченням**, а також **пропозиційними (булевими) зв'язками.**

Приклад. **Булеан множини.** Прикладом булевої алгебри може служити множина $B(U)$ усіх підмножин деякої множини U зі звичайними операціями об'єднання, перетину і доповнення у множині $B(U)$. Роль нуля у цьому випадку відіграє пуста множина, а одиниці — уся множина U. Дійсно, перелічені операції над множинами задовольняють законам M1) — M5), які є законами булевої алгебри. ♠

2.3.1. Алгебра булевих функцій

Алфавіт $X = \{x_1, x_2, \ldots, x_n\}$ називатимемо *алфавітом булевих змінних*, якщо кожна змінна $x_i \in X$ ($i = 1, 2,..., n$) може приймати лише одне з двох значень 0 або 1, тобто область інтерпретації змінних із множини X є множина $\{0, 1\}$. Константи 0 і 1 називають у цьому випадку **булевими константами.**

Визначення 48. *m-арна функція $f: X^m \to \{0, 1\}$ називається* **булевою функцією над алфавітом X**, *де $X = \{x_1, x_2,..., x_n\}$ — алфавіт булевих змінних.*

З визначення булевої функції від m аргументів випливає, що її область визначення скінченна і складається в точності з 2^m елементів. Довільний такий елемент (який називають *набором*) — це вектор довжини m, кожна компонента якого дорівнює 0 або 1. Виходячи з того, що довільна булева функція теж приймає лише одне з двох значень (0 або 1), можна підрахувати число всіх різних m-арних бу-

левих функцій. Дійсно, оскільки на кожному наборі з області визначення m-арної функції вона може приймати значення 0 або 1 незалежно від значень на інших наборах, то поступово розширюючи область визначення з одного набору до всіх 2^m наборів, щоразу подвоюється число різних функцій. Таким чином, існує 2 різні булеві функції від m змінних. Зокрема, існує 4 булеві функції однієї змінної, 16 булевих функцій двох змінних і т.д.. Завдяки тому, що області зміни аргументів і області значень булевих функцій збігаються, можна будувати суперпозиції булевих функцій, виконуючи підстановки одних із них на місце інших. У зв'язку з цим довільна булева функція f може розглядатися як операція над множиною всіх булевих функцій. Вибравши деяку множину функцій та множину основних операцій, можна будувати різні алгебри булевих функцій. При цьому вирази в такій алгебрі будуть представляти відповідні булеві функції. Наприклад, вираз $S(x_1, x_2, \ldots, x_k)$ представляє функцію $f(x_1, x_2, \ldots, x_k)$, якщо на довільному наборі значень змінних $x_1, x_2, \ldots, x_k$ значення S і f будуть збігатися. Особливе місце, як ми переконаємося пізніше, займають уже відомі нам булеві операції (функції) вигляду

$$f(x) = \neg x,\ g(x, y) = x \vee y,\ h(x, y) = x \wedge y.$$

Набір операцій над заданим алфавітом булевих змінних $X = \{x_1, x_2, \ldots, x_n\}$ називають **функціонально повним**, якщо довільна булева функція над алфавітом X може бути представлена хоча б одним виразом в алгебрі, яка визначається цим набором операцій. Закони булевої алгебри показують, що булева функція може мати декілька виразів, які її представляють. У зв'язку з цим виникає питання про канонічну форму виразів булевої алгебри. Розглянемо дві такі форми — *кон'юнктивну* і *диз'юнктивну*. Наведемо необхідні визначення.

Нехай $X = \{x_1, x_2, \ldots, x_n\}$ — алфавіт булевих змінних, а фіксованими булевими операціями є операції диз'юнкції, кон'юнкції і заперечення.

Елементарним добутком називається вираз, який являє собою кон'юнкцію довільного скінченного числа попарно різних символів булевих змінних з X, частина яких (можливо пуста) знаходиться під знаком заперечення. Наприклад, $\neg x \wedge y \wedge z$, $\neg x, x \wedge \neg y$, де $x, y, z \in X$, є елементарними добутками. До елементарних добутків також відносять і константу 1, про яку говорять, що вона складається з пустої множини членів.

Диз'юнктивною нормальною формою (ДНФ) називається диз'юнкція довільної скінченної множини попарно різних елементарних добутків. Це визначення включає в себе як випадок пустої

множини членів, так і випадок одного члена — булевої константи. Основну властивість ДНФ дає таке твердження.

Теорема 45. *Довільний вираз булевої алгебри можна перетворити до еквівалентної йому диз'юнктивної нормальної форми.*

Доведення теореми, по суті, є алгоритмом побудови ДНФ для заданого виразу. Опишемо його у вигляді кроків, які необхідно виконати в процесі побудови ДНФ. Кожний з цих кроків досить зрозумілий, і тому обмежимося лише їх послідовністю.

1. За допомогою законів де Моргана вираз перетворюється до вигляду, в якому немає заперечень ні перед диз'юнкцією, ні перед кон'юнкцією підвиразів даного виразу.

2. Застосовуємо закон подвійного заперечення до виразу, одержаного на кроці 1.

3. Застосовуємо закони дистрибутивності і закон комутативності для кон'юнкції до тих пір, поки вираз, який був одержаний на кроці 2, не набуде вигляду диз'юнкції кількох добутків.

4. Приводимо кожний диз'юнктивний член або до елементарного добутку, або до константи.

5. Застосовуючи закони для нуля і одиниці, спрощуємо одержаний вираз. Це і є шукана ДНФ. ■

Приклад. Побудувати ДНФ для виразу $\neg(x \wedge \neg y) \wedge ((x \wedge z) \vee \vee \neg(y \vee z))$. Застосовуючи двічі закони де Моргана, даний вираз можна перетворити до такого виразу

$$\neg(x \wedge \neg y) \wedge ((x \wedge z) \vee \neg(y \vee z)) = (\neg x \vee \neg\neg y) \wedge ((x \wedge z) \vee (\neg y \wedge \neg z)).$$

Застосовуючи закон подвійного заперечення, одержуємо таку формулу

$$(\neg x \vee y) \wedge ((x \wedge z) \vee (\neg y \wedge \neg z)).$$

Розкриваючи дужки за допомогою законів дистрибутивності, маємо $(\neg x \vee y) \wedge ((x \wedge z) \vee (\neg y \wedge \neg z)) = (\neg x \vee y) \wedge (x \wedge z) \vee ((\neg x \vee y) \wedge (\neg y \wedge \neg z)) = = (x \wedge z) \wedge (\neg x \vee y) \vee (\neg y \wedge \neg z) \wedge (\neg x \wedge y) = (x \wedge z \wedge \neg x) \vee (x \wedge y \wedge z) \vee \vee (\neg x \wedge \neg y \wedge \neg z) \vee (y \vee \neg y \wedge \neg z)$.

Нарешті, скориставшись законами $x \wedge \neg x = 0$, $0 \wedge x = 0$ і $0 \vee x = x$, остаточно знаходимо, що $\neg(x \wedge \neg y) \wedge ((x \wedge z) \vee \neg(y \vee z)) = (x \wedge y \wedge z) \vee \vee (\neg x \wedge \neg y \wedge \neg z)$. ♠

Виникає питання: чи буде ДНФ для заданої булевої функції єдиним її представленням, чи ні? Відповідь у загальному випадку негативна. Дійсно, наприклад ліві і праві частини закону поглинання задовольняють визначення ДНФ, але відрізняються одна від одної.

Можна ввести умови, у разі виконання яких нормальна форма булевої функції буде єдиною. Однією з таких форм є досконала ДНФ.

Елементарний добуток називається *конституентою одиниці* для алфавіту булевих змінних X, якщо він включає (під знаком заперечення, чи без нього) усі змінні з X. Зрозуміло, що кожна конституента одиниці набуває значення 1 лише на єдиному наборі із 2^m наборів для булевих змінних.

Визначення 49. *ДНФ булевої функції $f(x_1, x_2,..., x_n)$ називається досконалою ДНФ (ДДНФ), якщо всі її елементарні добутки є конституентами одиниці для множини $\{x_1, x_2,..., x_n\}$ аргументів функції.*

Теорема 46. *Довільний вираз булевої алгебри може бути перетворений до еквівалентної йому ДДНФ.*

Доведення. Нехай $S(x_1, x_2, \ldots, x_n)$ — деякий вираз булевої алгебри і $F(x_1, x_2, \ldots, x_n)$ — ДНФ цього виразу. Розглянемо довільний елементарний добуток виразу F (наприклад, $x_1 \wedge x_2 \wedge \ldots \wedge x_k$, $k = 1, 2,..., n$). Якщо $k = n$, то даний добуток є конституентою одиниці, і в цьому випадку переходимо до наступного елементарного добутку. Якщо $k < n$, то, використовуючи закон $x \vee \neg x = 1$, замінюємо $x_1 \wedge x_2 \wedge \ldots \wedge x_k$ на $x_1 \wedge x_2 \wedge \ldots \wedge x_k \wedge (x_{k+1} \vee \neg(x_{k+1}))$ і застосовуємо відповідні закони дистрибутивності. Цей процес продовжується до тих пір, поки всі елементарні добутки не будуть перетворені на конституенти одиниці для множини $\{x_1, x_2, \ldots, x_n\}$ змінних виразу S. ■

Теорема 47. *Довільна булева функція може бути представлена єдиною (з точністю до перестановки диз'юнктивних членів і їх множників) ДДНФ.*

Доведення. Нехай $f(x_1, x_2, \ldots, x_n)$ — довільна n-арна булева функція, а $(a_1, a_2,..., a_n)$ — довільний набір з області визначення функції f, на якому вона набуває значення 1. Позначимо $K(a_1, a_2,..., a_n) = \tilde{x}_1 \wedge \tilde{x}_2 \wedge \ldots \wedge \tilde{x}_n$ конституенту одиниці, множники якої визначаються за таким правилом:

$$\tilde{x}_i = \begin{cases} x_i, & \text{якщо } a_i = 1, \\ \neg x_i, & \text{якщо } a_i = 0. \end{cases}$$

Побудовану таким чином конституенту будемо називати конституентою, яка відповідає набору $(a_1, a_2, \ldots, a_n)$. Очевидно, що із 2^n конституент одиниці, тільки вона (з точністю до перестановки множників) буде дорівнювати 1 на наборі $(a_1, a_2,..., a_n)$. Звідси випливає, що ДДНФ, яка складена із усіх конституент одиниці, що відповіда-

ють наборам, де функція f набуває значення 1, представляє булеву функцію f. Зрозуміло також, що довільна зміна в складі конституент призводить до ДДНФ, яка вже не буде представляти цю булеву функцію. ■

Наслідок 20. *Довільна булева функція може бути представлена виразом булевої алгебри, тобто система операцій* $\vee$, $\wedge$ *і* $\neg$ *є функціонально повною.*

Підкреслимо, що згідно із законами де Моргана, функціонально повними будуть і класи функцій $\vee$, $\neg$ і $\wedge$, $\neg$.

Наслідок 21. *Для довільних двох виразів булевої алгебри можна перевірити їх еквівалентність.*

Дійсно, якщо вирази булевої алгебри S і S' еквівалентні, то кожний з них відповідно до теорем 45—47 перетворюється до однієї й тієї ж ДДНФ. Залишається перевірити чи це одна й та сама ДДНФ, чи ні. ■

Слід підкреслити той факт, що ДДНФ заданої функції може являти собою досить громіздку конструкцію. Наприклад, функція $f(x, y, z) = x \vee y \vee z$ має сім наборів (крім набору (0, 0, 0)), на яких вона дорівнює одиниці. Отже, її ДДНФ включає сім конституент одиниці.

2.3.2. Принцип двоїстості

Функція $f^*(x_1, x_2, \ldots, x_n) = \neg f(\neg x_1, \neg x_2, \ldots, \neg x_n)$ називається двоїстою до функції $f(x_1, x_2, \ldots, x_n)$. З даного визначення і законів булевої алгебри випливає, що функція двоїста до двоїстої функції, збігається з висхідною, тобто $(f^*)^* = f$. Легко переконатися, що диз'юнкція і кон'юнкція, а також 0 і 1 — двоїсті одна до одної функції, а функція заперечення двоїста до самої себе. Дійсно,

$$(\neg x)^* = \neg(\neg\neg x) = \neg x;$$
$$(0)^* = \neg(0) = 1 \text{ і } (1)^* = \neg(1) = 0;$$
$$(x \vee y)^* = \neg(\neg x \vee \neg y) = \neg\neg x \wedge \neg\neg y = x \wedge y;$$
$$(x \wedge y)^* = \neg(\neg x \wedge \neg y) = \neg\neg x \vee \neg\neg y = x \vee y.$$

Теорема 48 (*Закон двоїстості*). *Нехай*

$$F(x_1, x_2, \ldots, x_n) = f(f_1(x_1, x_2, \ldots, x_n), f_2(x_1, x_2, \ldots, x_n), \ldots, f_m(x_1, x_2, \ldots, x_n)),$$

тоді

$$F^*(x_1, x_2, \ldots, x_n) = f^*(f_1^*(x_1, x_2, \ldots, x_n), f_2^*(x_1, x_2, \ldots, x_n), \ldots,$$
$$f_m^*(x_1, x_2, \ldots, x_n)),$$

де $F^*, f^*, f_1^*, f_2^*, \ldots, f_m^*$ *— функції двоїсті до функцій* $F, f, f_1 f_2, \ldots, f_m$ *відповідно.*

Доведення випливає з таких рівностей:

$F^*(x_1,..., x_n) = \neg F(\neg x_1,...,\neg x_n) =$

$= \neg f(f_1(\neg x_1,\ldots, \neg x_n), f_2(\neg x_1, ..., \neg x_n), ..., f_m(\neg x_1, ..., \neg x_n)) =$

$= \neg f(\neg\neg f_1(\neg x_1,..., \neg x_n), \neg\neg f_2(\neg x_1,..., \neg x_n),..., \neg\neg f_m(\neg x_1,..., \neg x_n)) =$

$= \neg f(\neg f_1^*(x_1, ..., x_n), \neg f_2^*(x_1,..., x_n),..., \neg f_m^*(x_1,..., x_n)) =$

$= f^*(f_1^*(x_1, ..., x_n), f_2^*(x_1,..., x_n),..., f_m^*(x_1,..., x_n)).$ ■

Виходячи із закону двоїстості для констант 0 і 1, операцій диз'юнкції і кон'юнкції, а також заперечення, одержуємо, що в булевій алгебрі функцій вираз S^*, двоїстий до виразу S, є результатом одночасної заміни у виразі S усіх диз'юнкцій на кон'юнкції, усіх кон'юнкцій на диз'юнкції, усіх нулів на одиниці і всіх одиниць на нулі. Користуючись поняттям двоїстого виразу, можна перейти від уже розглянутого представлення булевих функцій у вигляді ДНФ до другого відомого представлення — **кон'юнктивної нормальної форми (КНФ).** При цьому двоїстим до поняття елементарного добутку буде поняття елементарної диз'юнкції, двоїстим до поняття конституенти одиниці — поняття конституенти нуля.

2.3.3. Алгебра Жегалкіна

Алгебра Жегалкіна — це множина булевих функцій, на якій визначені такі операції:

— нульарна операція 1,

— бінарна операція кон'юнкція ($\wedge$),

— бінарна операція суми за модулем два ($\oplus$).

Користуючись операціями додавання за модулем два і одиниці, можна ввести константу 0, тобто

$$1 \oplus 1 = 0.$$

Основними тотожними співвідношеннями даної алгебри є такі:

ж1) $x \wedge y = y \wedge x$; $x \wedge x = x$ $x \oplus y = y \oplus x$;

ж2) $x \oplus (y \oplus z) = (x \oplus y) \oplus z$;

$x \wedge (y \wedge z) = (x \wedge y) \wedge z$;

ж3) $x \oplus x = 0$;

ж4) $x \oplus 0 = x$;

ж5) $x(y \oplus z) = x \wedge y \oplus x \wedge z$.

Операції заперечення і диз'юнкції в цій алгебрі вводяться за допомогою таких співвідношень:

$$\neg x = x \oplus 1; x \vee y = x \wedge y \oplus x \oplus y.$$

Користуючись цими співвідношеннями, довільну ДДНФ можна перетворити до виразу, в якому відсутні операції заперечення і

диз'юнкції. Після цього, застосовуючи закони ж1) — ж5) і приводячи подібні члени, одержуємо представлення булевої функції в алгебрі Жегалкіна, яке називається **поліномом Жегалкіна.** Для однозначності задання булевої функції поліномом Жегалкіна користуються так званим канонічним поліномом.

Канонічним поліномом називається скінченна сума таких попарно різних добутків змінних, де в одному добутку ніяка змінна не трапляється більше одного разу. При цьому до числа добутків відносять добутки, які складаються як із одного співмножника (окрема змінна), так і з пустої множини співмножників (константа 1).

Приклад. Вирази $x \wedge y \wedge z \oplus x \wedge z$, y, 1, очевидно, є канонічними поліномами, а вирази $z \wedge x \wedge z$, $x \wedge z \oplus z \wedge x$ — ні, оскільки в першому з них повторюється множник z, а в другому — рівні між собою доданки. ♠

Теорема 49. *Довільна булева функція може бути представлена канонічним поліномом Жегалкіна, причому це представлення єдине з точністю до перестановки доданків та їх співмножників.*

Доведення. Нехай $f(x_1, x_2,..., x_n)$ — довільна булева функція і a_1, $a_2,..., a_m$ $(m = 2^n)$ — така послідовність попарно різних наборів з області визначення цієї функції, що у довільних двох наборів, які стоять поряд у цій послідовності, число одиниць у другому або таке ж, як і в першому, або на одиницю більше. Множину, яку складають перші k $(1 \le k \le 2^n)$ членів цієї послідовності, позначимо через G_k. Побудуємо таку послідовність $F_1, F_2, \ldots, F_k$ канонічних поліномів змінних $x_1, x_2, \ldots, x_n$, що значення $F_k(x_1,..., x_n)$ і $f(x_1,..., x_n)$ збігаються на множині G_k і поліноми F_k і F_{k+1} або збігаються або другий одержано з першого шляхом додання ще одного доданка. Якщо така послідовність побудована, то $F_m(x_1,..., x_n)$, очевидно, буде представляти саму функцію $f(x_1,..., x_n)$. Виберемо F_1 як константу $f(0,..., 0)$.

Припустимо, що вже побудовані перші k $(k < 2^n)$ поліномів шуканої послідовності. Візьмемо набір a_{k+1}, який розрізняє між собою множини G_k і G_{k+1}, і порівняємо значення $F_k(a_{k+1})$ і $f(a_{k+1})$. Якщо вони збігаються, то за F_{k+1} може бути вибраний поліном F_k. Якщо ж ці значення відрізняються одне від одного, то будуємо добуток P_{k+1}, який включає як співмножники ті і тільки ті змінні із $x_1,..., x_n$, які на наборі a_{k+1} дають одиницю, і покладемо $F_{k+1} = P_{k+1} \oplus F_k$. Для набору a_{k+1} значення висхідної функції f і поліному F_{k+1} збігаються за побудовою. Для наборів із G_k збіг значень f і F_{k+1} буде в силу того, що на них добуток P_{k+1} перетворюватиметься на нуль.

Дійсно, оскільки довільний набір із G_k відмінний від a_{k+i} і включає не більше одиниць, ніж число співмножників у добутку P_{k+i}, то для довільного такого набору хоча б один співмножник із P_{k+i} буде дорівнювати нулю. Отже, існування канонічного поліному, який представляє функцію $f(x_1,..., x_n)$, доведено.

Покажемо тепер єдиність цього поліному. Нехай F_1 і F_2 — поліноми, які представляють одну й ту саму функцію f і відрізняються не тільки порядком переліку доданків і співмножників. Розглянемо поліном $F = F_1 \oplus F_2$, в якому відповідно до тотожності $x \oplus x = 0$ виконується зведення подібних доданків. Поліном F відповідно до припущення про вибір F_1 і F_2 повинен відрізнятися від константи 0, а з другого боку, в силу того, що F_1 і F_2 представляють одну й ту саму функцію, він повинен на всіх наборах з області визначення перетворюватися на нуль. Але ці умови суперечать одна одній. Дійсно, якщо поліном F має доданок 1, то на наборі (0, 0,..., 0) він буде відмінним від 0, у протилежному випадку виберемо в F доданок з мінімальним числом співмножників і побудуємо набір, на якому всі доданки, окрім вибраного, перетворюються на нуль. ■

З доведеної теореми випливає, що система булевих функцій, яка складається з константи 1, кон'юнкції і суми за модулем два, як і розглянута вище система $\wedge$, $\vee$, $\neg$, є функціонально повною.

2.3.4. Класи Поста

Розглянемо тепер питання загального характеру: нехай задана довільна система булевих функцій, яким умовам повинна задовольняти ця система, щоб бути функціонально повною. Ці умови називаються умовами повноти. Пост, характеризуючи умови повноти довільної системи булевих функцій, виділив п'ять класів булевих функцій, які замкнуті відносно суперпозиції:

(1) функції, які зберігають 0;
(2) функції, які зберігають 1;
(3) функції, двоїсті самі до себе;
(4) лінійні функції;
(5) монотонні функції.

Визначення 50. *Функція $f(x_1,..., x_n)$ називається функцією, яка* **зберігає 0**, *якщо $f(0,...,0) = 0$.*
Функція $f(x_1,..., x_n)$ називається функцією, яка **зберігає 1**, *якщо $f(1,...,1) = 1$.*
Функція, двоїста сама до себе, називається **самодвоїстою** *функцією.*

Функція називається **лінійною,** *якщо кожний елементарний добуток канонічного поліному Жегалкіна, що представляє цю функцію, має не більше одного співмножника.*
Функція називається **монотонною,** *якщо із того, що вона набуває значення* 1 *на деякому наборі a, випливає, що вона набуває значення* 1 *на довільному наборі b, який одержується із набору a шляхом заміни довільного числа нулів на одиниці.*

Замкнутість кожного із п'яти класів Поста випливає із визначення функцій, які входять до цих класів. Для функцій, які є твірними розглянутих вище алгебр, належність їх до того чи іншого класу Поста наводиться в табл. 6.

Таблиця 6

Функції	$\vee$	$\wedge$	$\neg$	$\oplus$
Які зберігають 0	+	+	–	+
Які зберігають 1	+	+	–	–
Самодвоїсті	–	–	+	–
Лінійні	–	–	+	+
Монотонні	+	+	–	–

+ — функція належить класу Поста;
– — функція не належить класу Поста.

З таблиці 6, зокрема, видно, що кожний із п'яти класів Поста є власною підмножиною всієї множини булевих функцій.

Теорема 50 (*Теорема Поста про функціональну повноту булевих функцій*). *Для того щоб система булевих функцій S була функціонально повною над алфавітом булевих змінних* $X = \{x_1, x_2, \ldots, x_n\}$, *необхідно і достатньо, щоб ця система включала хоча б одну функцію, яка не зберігає* 0, *хоча б одну функцію, яка не зберігає* 1, *хоча б одну несамодвоїсту функцію, хоча б одну нелінійну функцію і хоча б одну немонотонну функцію.*

Доведення. Необхідність умов теореми випливає з того, що кожний із класів Поста є власною підмножиною всієї множини булевих функцій. Для доведення достатності зафіксуємо п'ять функцій:

- $f_1(x_1, \ldots, x_n)$ $(n \geq 1)$, яка не зберігає 0;
- $f_2(x_1, \ldots, x_n)$ $(n \geq 1)$, яка не зберігає 1;
- $f_3(x_1, \ldots, x_n)$ $(n \geq 1)$, яка несамодвоїста;

• $f_4(x_1,..., x_n)$ $(n \geq 1)$, яка нелінійна;
• $f_5(x_1,..., x_n)$ $(n \geq 1)$, яка немонотонна.

Зауважимо, що деякі з цих функцій можуть збігатися. Відповідно до наслідку 20 достатньо показати, що з вибраних п'яти функцій будується, наприклад, кон'юнкція і заперечення.

Покажемо, насамперед, як побудувати константи 0 і 1 у такій алгебрі. Підставляючи у функцію f_1 замість усіх змінних одну й ту саму змінну x, отримуємо функцію $h(x) = f_1(x, x,..., x)$, для якої за умовою справедлива рівність $h(0) = 1$. Якщо виявиться, що $h(1) = 1$, то $h(x)$ — функція, тотожно рівна 1. Підставляючи її замість аргументів у функцію f_2, отримуємо другу константу — 0. Якщо ж таким шляхом не вдається одержати 0 і 1, то функція $h(x)$ є запереченням.

Лема 3. *Якщо система S включає несамодвоїсту функцію і функцію заперечення, то серед її суперпозицій існують також обидві константи* — 0 *і* 1.

Доведення. Дійсно, якщо $f(x_1,..., x_n)$ — несамодвоїста функція, то існує такий набір $(a_1, ..., a_n)$ її аргументів, що $f(a_1,..., a_n) = f(\neg a_1,..., \neg a_n)$. Але тоді для $i = 1, 2,..., n$, підставляючи замість змінної x_i змінну x, якщо $a_i = 1$, і її заперечення $\neg x$, якщо $a_i = 0$, отримуємо функцію $h(x)$, яка незалежно від значення змінної x буде приймати одне й те саме значення, тобто буде константою. Якщо цю константу тепер підставити у функцію заперечення, то отримаємо другу константу. ■

Лема 4. *Якщо система S включає немонотонну функцію і обидві константи* 0 *і* 1, *то серед її суперпозицій існує функція заперечення.*

Доведення. Нехай $f(x_1,..., x_n)$ немонотонна функція, тоді знайдуться такі набори a і a', що функція $f(a) = 1, f(a') = 0$, і ті компоненти, якими ці набори відрізняються, у наборі a дорівнюють 0, а у наборі a' – 1. Неважко побудувати таку послідовність $a = a_1, a_2,..., a_k = a'$, що довільні два сусідні набори в цій послідовності відрізняються один від одного лише однією компонентою, яка дорівнює 0 для першого з них і 1 — для другого. Після цього розглянемо послідовність значень $f(a_1) = 1, f(a_2),..., f(a_k) = 0$ і знайдемо в послідовності наборів такі два набори a_i і a_{i+1}, що $f(a_i) = 1$ і $f(a_{i+1}) = 0$. Нехай j — номер компоненти, за якою відрізняються ці набори. Підставивши у функцію $f(x_1,..., x_n)$ замість змінної x_j змінну x, а замість решти змінних відповідні їм константи 0 і 1 з наборів a_i і a_{i+1}, отримаємо функцію $h(x)$, яка, очевидно, буде запереченням. ■

Для остаточного доведення теореми тепер необхідно показати, як можна сконструювати кон'юнкцію $x_1 \wedge x_2$. Для цього візьмемо нелі-

нійну функцію f_4. Її канонічний поліном включає хоча б один доданок, який складається більше ніж із двох змінних (нехай для визначеності це будуть змінні x_1 і x_2). Користуючись дистрибутивністю кон'юнкції відносно додавання за модулем два, функцію f_4 можна представити у вигляді $x_1 \wedge x_2 \wedge g_1(x_3,..., x_n) \; x_1 \wedge g_2(x_3,..., x_n) \oplus x_2 \wedge g_3(x_3,..., x_n) \oplus g_4(x_3,..., x_n)$, де g_1 тотожно не дорівнює нулю. Підставляючи відповідним чином константи замість змінних $x_3,..., x_n$, функцію f_4 перетворимо до вигляду $x_1 \wedge x_2 \oplus a \wedge x_1 \oplus b \wedge x_2 \oplus c$, де a, b, c — константи 0 або 1. Підставимо тепер замість x_1 функцію $x_1 \oplus b$ ($x_1 \oplus b = x_1$, якщо $b = 0$, або $\neg x_1$, якщо $b = 1$), і замість x_2 — функцію $x_2 \oplus a$, отримаємо функцію

$$(x_1 \oplus b) \wedge (x_2 \oplus a) \oplus a \wedge (x_1 \oplus b) \oplus b \wedge (x_2 \oplus a) \oplus c =$$
$$= x_1 \wedge x_2 \oplus a \wedge x_1 \oplus b \wedge x_2 \oplus a \wedge b \oplus a \wedge x_1 \oplus a \wedge b \oplus b \wedge x_2 \oplus a \wedge b \oplus c =$$
$$= x_1 \wedge x_2 \oplus (a \wedge b \oplus c).$$

Якщо тепер константа $a \wedge b \oplus c = 0$, то отримана функція шукана, якщо ж $a \wedge b \oplus c = 1$, то шуканою функцією буде її заперечення. ■

2.4. Ґратки (структури)

Визначення 51. *Алгебра* $G = (A, \Omega) \in K(\Omega, Eq)$ *називається* **граткою** *або* **решіткою**, *якщо* Ω *складається з двох бінарних операцій* $\vee$ (*верхня грань*) *і* $\wedge$ (*нижня грань*), *і для довільних* $a, b, c \in A$ *справедливі такі співвідношення*:
C1) $a \vee a = a \wedge a = a$ — *ідемпотентність*;
C2) $a \vee b = b \vee a$, $a \wedge b = b \wedge a$ — *комутативність*;
C3) $a \vee (b \vee c) = (a \vee b) \vee c$,
$a \wedge (b \wedge c) = (a \wedge b) \wedge c$ — *асоціативність*;
C4) $a \vee (a \wedge b) = a$, $a \wedge (a \vee b) = a$ — *поглинання*.
Якщо на множині A *задана лише операція* $\vee$ ($\wedge$), *що задовольняє законам C1) — C3), то алгебра* $G(A, \Omega)$ *називається* **верхньою (нижньою) напівграткою.**

Якщо алгебра являє собою напівґратку, то на її носії можна задати відношення часткового порядку, яке має ту властивість, що для довільних елементів $a, b \in A$ існує елемент $c \in A$ такий, що $a \le c$ і $b \le c$ ($a \ge c$ і $b \ge c$) (самі елементи a і b можуть бути такими, що не порівнюються між собою).

Дійсно, нехай $G = (A, \Omega)$ — верхня напівґратка. Визначимо для довільних $a, b \in A$: $a \le b \Leftrightarrow a \vee b = b$. Покажемо, що $\le$ — частковий

порядок на A. Для цього потрібно показати, що $\leq$ рефлексивне, антисиметричне і транзитивне.

Рефлексивність. $a \leq a \Leftrightarrow a \vee a = a$, але останнє є *C1*.

Антисиметричність: $a \leq b \Leftrightarrow a \vee b = b$, $b \leq a \Leftrightarrow b \vee a = a$ і в силу комутативності операції $\vee$ (*C2*) заключаємо, що $a = b$.

Транзитивність: $a \leq b \Leftrightarrow a \vee b = b$, $b \leq c \Leftrightarrow b \vee c = c \Rightarrow (a \vee b) \vee c = a \vee (b \vee c) = a \vee c = c \Rightarrow a \vee c = c$, звідки випливає, що $a \leq c$.

Таким чином, ґратки можна вважати частково упорядкованими множинами.

Якщо G — ґратка, то із *C1*) — *C4*) безпосередньо випливають такі властивості.

Твердження 9. *Для довільних елементів $a, b, c \in G$ істинно*

(1) $a \wedge b \leq a$;

(2) $a \leq b$ *тоді і тільки тоді, коли* $a \wedge b = a$;

(3) якщо $a \leq b$ *і* $c \leq d$, *то* $a \vee c \leq b \vee d$ *і* $a \wedge c \leq b \wedge d$;

(4) $(a \wedge c) \vee (b \wedge c) \leq c$.

Доведення. (1) випливає безпосередньо із *C4*). Дійсно, оскільки $a \vee (a \wedge b) = a$, то за визначенням відношення $\leq$ маємо $a \wedge b \leq a$.

(2) Якщо $a \leq b$, то $a \vee b = b$ і тоді із *C4*) отримуємо $a \wedge (a \vee b) = a \wedge b = a$.

Навпаки, якщо $a \wedge b = a$, і оскільки $b \vee (b \wedge a) = b \vee a = b$, то це означає, що $a \leq b$.

(3) Якщо $a \leq b$ і $c \leq d$, то це означає, що $a \vee b = b$ і $c \vee d = d$. Але тоді, користуючись *C2*) і *C3*), отримуємо $(a \vee c) \vee (b \vee d) = (a \vee b) \vee (c \vee d) = b \vee d$, тобто $(a \vee c) \leq (b \vee d)$. Аналогічно отримуємо $(a \wedge c) \wedge (b \wedge d) = (a \wedge b) \wedge (c \wedge d) = a \wedge c$, тобто $(a \wedge c) \leq (b \wedge d)$ на основі (2).

(4) На основі (1) маємо, що $a \wedge c \leq c$ і $b \wedge c \leq c$. Але тоді $(a \wedge c) \vee (b \wedge c) \leq c \vee c = c$ на основі (3) і *C1*). ■

Інші корисні нерівності для елементів ґратки наведені у вправах у кінці розділу (див. вправу 20).

Часто можна зустріти й таке визначення ґратки [21, 34].

Частково упорядкована множина A відношенням $\leq$ називається *ґраткою*, якщо вона задовольняє таким умовам:

(i) для довільних елементів a, b із A існує елемент $c = a \wedge b$ — *перетин елементів* a і b — такий, що $c \leq a$ і $c \leq b$, причому якщо c' — деякий елемент із A, такий що $c' \leq a$ і $c' \leq b$, то $c' \leq c$;

(ii) для довільних елементів a, b із A існує елемент $d = a \vee b$ — *об'єднання елементів* a і b — такий, що $d \geq a$ і $d \geq b$, причому якщо d' — деякий елемент із A, такий, що $d' \geq a$ і $d' \geq b$, то $d' \geq d$.

З цього визначення випливає, що перетин і об'єднання елементів у A визначені однозначно, тобто довільна ґратка є універсальною алгеброю з двома бінарними операціями $\wedge$ і $\vee$.

Нехай $a \le b$, $a, b \in A$. Розглянемо чому дорівнюють однозначно визначені елементи $a \wedge b$ і $a \vee b$. В силу рефлексивності відношення часткового порядку $\le$ таким елементом є елемент a, оскільки якщо $a = a \wedge b$, то $a \le a$ і $a \le b$. Аналогічно знаходимо, що $a \vee b = b$. Отже, відношення $\le$ може бути задане за допомогою однієї із цих операцій. Має місце

Твердження 10. *Якщо A — ґратка, визначена за допомогою (i), (ii), то алгебра $G = (A, \{\wedge, \vee\})$ задовольняє законам C1) — C4), і навпаки, якщо $G = (A, \{\wedge, \vee\})$ — алгебра, в якій виконуються закони C1) — C4), то множина A є ґраткою відповідно до пунктів визначення (i), (ii).*

Доведення. Закони *C1*) і *C2*) у ґратці A очевидні. Розглянемо *C3*). Покажемо спочатку, що коли $a \le b$ і $c \le d$, то $a \wedge c \le b \wedge d$ і $a \vee c \le b \vee d$. Дійсно,

$$a \wedge c = (a \wedge b) \wedge (c \wedge d) \le a \wedge b \le b,$$
$$a \wedge c = (a \wedge b) \wedge (c \wedge d) \le c \wedge d \le d.$$

Але тоді, якщо $d' = b \wedge d$, то $d' \le b$, $d' \le d$ і в силу (i) $a \wedge c \le d'$, тобто $a \wedge c \le b \wedge d$. Аналогічно доводиться і друга нерівність. Тепер, користуючись доведеними нерівностями, отримаємо

$$(a \wedge b) \wedge c \le a \wedge b \le a,$$
$$(a \wedge b) \wedge c \le a \wedge b \le b,$$
$$(a \wedge b) \wedge c \le c,$$

звідки $(a \wedge b) \wedge c \le a \wedge (b \wedge c)$. Аналогічно із

$$a \wedge (b \wedge c) \le a,$$
$$a \wedge (b \wedge c) \le b \wedge c \le b,$$
$$a \wedge (b \wedge c) \le b \wedge c \le c$$

отримуємо $a \wedge (b \wedge c) \le (a \wedge b) \wedge c$. В силу антисиметричності відношення $\le$ маємо $a \wedge (b \wedge c) = (a \wedge b) \wedge c$.

Розглянемо другий із законів *C4*). В силу (i) $a \wedge (a \vee b) \le a$. Але оскільки $a \le a$ і $a \le a \vee b$ в силу (ii), то $a = a \wedge a \le a \wedge (a \vee b)$, тобто другий закон поглинання має місце.

Інші деталі доведення пропонуються читачеві як прості вправи.■

Якщо в ґратці існують такі елементи, як **0 (нуль)** і **1 (одиниця)**, що для довільного елемента a ґратки G мають місце тотожності

$$a \vee 0 = a \wedge 1 = a, \quad a \wedge 0 = 0, a \vee 1 = 1,$$

то ґратка G називається ґраткою відповідно з нулем і одиницею.

Конкретним прикладом ґратки може служити булеан $B(U)$ деякої множини U. Частковий порядок на $B(U)$ вводиться за допомогою операції включення для підмножин, а роль операцій $\vee$ і $\wedge$ відіграють операції $\cup$ і $\cap$ відповідно.

Зауважимо, що множина $B(U)$ буде ґраткою з нулем і одиницею, роль яких виконують відповідно пуста множина $\varnothing$ і вся множина U.

Оскільки ґратка є частково упорядкованою множиною, то можна розглядати послідовності елементів $a_1, a_2, \ldots, a_n, \ldots$ із G таких, що

$$a_1 < a_2 < \ldots < a_n < \ldots \ (a_1 > a_2 > \ldots > a_n > \ldots). \qquad (17)$$

Послідовність (17) називається *строго зростаючою* (*строго спадаючою*) або просто *зростаючим* (*спадаючим*) *ланцюгом*.

Визначення 52. *Ґратка $G = (A, \Omega)$ називається ґраткою з* **умовою обриву спадаючих ланцюгів (з умовою обриву зростаючих ланцюгів)**, *якщо довільний зростаючий (спадаючий) ланцюг обривається на скінченному індексі. Тобто для довільного ланцюга* (17) *існує такий індекс n, починаючи з якого цей ланцюг* **стабілізується**, *тобто*

$$a_n = a_{n+1} = \ldots .$$

Ґратки з умовою обриву спадаючих ланцюгів є алгебрами, які мають цілий ряд цікавих властивостей.

Теорема 51. *Якщо алгебра $G = (A, \Omega)$ — ґратка з умовою обриву спадаючих ланцюгів, то довільна непуста підмножина множини A має хоча б один мінімальний у цій множині елемент.*

Доведення. Припустимо супротивне: існує деяка непуста підмножина A_1 множини A, яка не має мінімальних елементів. Позначимо по одному елементу в кожній непустій підмножині множини A_1, користуючись аксіомою вибору, а потім побудуємо послідовність елементів $a_1, a_2, \ldots, a_n, \ldots$.

Першим елементом a_1 беремо елемент, позначений у множині A_1, другим елементом — a_2, беремо елемент строго менший за a_1 у множині A_1, тобто $a_2 \in A_1 \setminus \{a_1\}$, $a_2 < a_1$, і т. д. Відповідно до припущення про відсутність мінімального елемента у множині A_1, множина $A_1 \setminus \{a_1\}$ непуста, множина $A_1 \setminus \{a_1, a_2\}$ непуста і т. д. Побудована таким чином послідовність є нескінченним строго спадаючим ланцюгом, і оскільки цей ланцюг лежить у множині A, то алгебра G не може бути ґраткою з умовою обриву спадаючих ланцюгів. Отримана суперечність доводить теорему. ■

Теорема 52. *Якщо ґратка $G = (A, \Omega)$ має таку властивість, що довільна непуста підмножина із A має хоча б один мінімальний елемент, то в ґратці G виконується умова індуктивності.*

Доведення. Припустимо, що у множині A існує деяка підмножина елементів A_1 така, що елементи із A_1 не задовольняють властивості P. Зважаючи на існування мінімальних елементів, підмножина A_1 має хоча б один мінімальний елемент a. Елемент a не може бути мінімальним для всієї множини A, бо в протилежному випадку хибною буде посилка 1) умови індуктивності. Оскільки для всіх елементів $c < a$ умова P істинна, то згідно з другою посилкою умови індуктивності, $P(a)$ теж має бути істинним. Отримана суперечність доводить теорему. ■

Завдяки тому, що відношення часткового порядку двоїсте, то все сказане залишається справедливим і для ґраток з умовою обриву зростаючих ланцюгів.

2.4.1. Дистрибутивні і дедекіндові ґратки

Визначення 53. *Ґратка G називається* **дистрибутивною**, *якщо Eq, окрім C1) — C4) включає закон дистрибутивності, тобто для довільних її елементів a, b, c має місце*:

$$C5)\ (a \vee b) \wedge c = (a \wedge c) \vee (a \wedge c).$$

Зауважимо, що не всі ґратки є дистрибутивними, як і не всі ґратки задовольняють умові максимальності (мінімальності) і не обов'язково повинні мати 0 або 1, або і одне й друге разом.

Визначення 54. *Ґратка G називається* **дедекіндовою**, *або* **модулярною**, *якщо, окрім законів C1) — C4), для довільних її елементів a, b, c, таких що $a \le c$, виконується модулярний закон*:

$$M3)\ (a \vee b) \wedge c = a \vee (b \wedge c).$$

Зробимо декілька зауважень відносно дистрибутивних і дедекіндових ґраток.

Для дистрибутивних ґраток має місце така теорема.

Теорема 53. *Наступні властивості ґратки G еквівалентні*:

(1) G — дистрибутивна;

(2) $(a \wedge b) \vee c = (a \vee c) \wedge (b \vee c)$ для довільних a, b, c із G;

(3) $(a \wedge b) \vee (b \wedge c) \vee (c \wedge a) = (a \vee b) \wedge (b \vee c) \wedge (c \vee a)$ для довільних a, b, c із G;

(4) якщо для деякого c із G мають місце рівності $a \vee c = b \vee c$ і $a \wedge c = b \wedge c$, то $a = b$.

Доведення. (1) $\Rightarrow$ (2). Маємо

$$(a \vee c) \wedge (b \vee c) = ((a \vee c) \wedge b) \vee c = (a \wedge b) \vee (b \wedge c) \vee c = (a \wedge b) \vee c.$$

Отже, має місце (2).

(2) $\Rightarrow$ (3). Дійсно,

$$(a \wedge b) \vee (b \wedge c) \vee (c \wedge a) = (a \vee (b \wedge c) \vee (c \wedge a)) \wedge (b \vee (b \wedge c) \vee (c \wedge a)) =$$
$$= (a \vee (b \wedge c)) \wedge (b \vee (c \wedge a)) = (b \vee a) \wedge (c \vee a) \wedge (b \vee c) \wedge (b \vee a) =$$
$$= (a \vee b) \wedge (b \vee c) \wedge (c \vee a).$$

(3) $\Rightarrow$ (1). Нехай G задовольняє умові (3), $a, b, c \in G$ і $a \leq c$. Тоді

$$(a \wedge c) \vee (b \wedge c) = (a \wedge b) \vee (b \wedge c) \vee (c \wedge a) =$$
$$= (a \vee b) \wedge (b \vee c) \wedge (c \vee a) = c \wedge (a \vee b) \wedge (b \vee c) = c \wedge (a \vee b).$$

Отже, у G виконується модулярний закон. Далі, припустимо $u = (a \wedge b) \vee (b \wedge c) \vee (c \wedge a)$, $v = (a \vee b) \wedge (b \vee c) \wedge (c \vee a)$. За умовою $u = v$ і, отже, $c \wedge u = c \wedge v$. Оскільки $(a \wedge c) \vee (b \wedge c) \leq c$, то, враховуючи уже доведений модулярний закон, отримуємо

$$c \wedge u = c \wedge ((a \wedge b) \vee (b \wedge c) \vee (c \wedge a)) =$$
$$= c \wedge ((b \wedge c) \vee (a \wedge c)) \vee (a \wedge b \wedge c) = (a \wedge c) \vee (b \wedge c).$$
$$c \wedge v = c \wedge ((a \vee b) \wedge (b \vee c) \wedge (c \vee a)) = c \wedge (a \vee b),$$

що й потрібно було довести.

(1) $\Rightarrow$ (4). Якщо $a \vee c = b \vee c$ і $a \wedge c = b \wedge c$, то застосовуючи (1), отримуємо

$$a = a \wedge (a \vee c) = a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c) =$$
$$= (a \wedge b) \vee (b \wedge c) = (a \vee c) \wedge b = (b \vee c) \wedge b = b.$$

(4) $\Rightarrow$ (3). Пропонується читачеві як вправа. ■

Умова (3) доведеної теореми показує, що частково упорядкована множина відносно двоїстого відношення часткового порядку теж буде дистрибутивною ґраткою. Отже, до дистрибутивних ґраток застосовний принцип двоїстості (відповідно до якого довільне твердження залишається справедливим, якщо в ньому замінити знак $\leq$ на двоїстий до нього $\geq$).

Розглянемо коротко дедекіндові ґратки.

Теорема 54. *Наступні властивості ґратки G еквівалентні:*

(1) G — дедекіндова ґратка;

(2) $a \wedge ((a \wedge b) \vee c) = (a \vee b) \vee (a \wedge c)$ для довільних елементів a, b, c із G;

(3) якщо $a \geq b$ і для деякого c із G мають місце рівності $a \vee c = b \vee c$ і $a \wedge c = b \wedge c$, то $a = b$.

Доведення. Нехай G — дедекіндова ґратка, тобто має місце (1). Тоді для довільних $a, b, c \in G$ маємо $a \wedge b \leq a$ і, отже,

$$a \wedge ((a \wedge b) \vee c) = ((a \wedge b) \vee c) \wedge a = (a \wedge b) \vee (a \wedge c).$$

Якщо ґратка G — дедекіндова і виконані умови (3), то

$$a = a \wedge (a \vee c) = a \wedge (b \vee c) = b \vee (a \wedge c) = b \vee (b \wedge c) = b.$$

Таким чином, зі справедливості умови (1) випливають умови (2) і (3).

Якщо виконано (2) і $a \leq c$, то маємо

$$(a \vee b) \wedge c = ((a \wedge c) \vee b) \wedge c = (a \wedge c) \vee (b \wedge c) = a \vee (b \wedge c),$$

тобто з умови (2) випливає (1).

Нарешті, нехай має місце (3). Якщо $a, b, c \in G$ і $a \leq c$, то

$$a \vee b \leq (a \vee b) \wedge c \leq a \vee c$$

і

$$b \wedge c \leq (a \vee (b \wedge c)) \wedge b \leq (c \vee (b \wedge c)) \wedge b = b \wedge c.$$

Звідки

$$((a \vee b) \wedge c) \vee b = a \vee b,$$
$$(a \vee (b \wedge c)) \vee b = a \vee b,$$
$$(a \vee b) \wedge c \wedge b = b \wedge c$$

і

$$(a \vee (b \wedge c)) \wedge b = b \wedge c.$$

Оскільки

$$(a \vee b) \wedge c \geq a \vee (b \wedge c),$$

то, застосовуючи рівності з умови (3), отримуємо

$$(a \vee b) \wedge c = a \vee (b \wedge c). \blacksquare$$

Теорема 55. *(Закон скорочення). Якщо a, b, c елементи дедекіндової ґратки і $(a \vee b) \wedge c = 0$, то $a \wedge (b \vee c) = a \wedge b$.*

Доведення пропонується як проста вправа.

На завершення зауважимо, що закон, двоїстий до закону модулярності, має вигляд

$$(a \wedge b) \vee c = a \vee (b \wedge c),$$

де $a \geq c$, тобто знову є модулярним законом. Таким чином, ґратка, двоїста до дедекіндової ґратки, сама буде дедекіндовою ґраткою. Отже, дедекіндові ґратки задовольняють принцип двоїстості.

2.5. Багатоосновні алгебри

Окрім алгебр з однією основною множиною доводиться розглядати алгебри, які задані на декількох різних носіях. Такі алгебри називають **багатоосновними** або **багатосортними** [21, 9, 17].

Нехай $U = \{A_i, i \in I\}$, де A_i — деякі множини, а I — множина сортів — імена відповідних A_i. Пару $U = \{A_i, i \in I\}$ називають *комплектом* або *набором*. В односортному випадку кожній операції відповідає її арність, а у многосортному випадку з кожною операцією пов'язують її тип. *Типом операції* ω називають послідовність елементів із I виду $s = (i_1, i_2, \ldots, i_n; j)$, а операцією ω типу s на комплекті $U = \{A_i, i \in I\}$ — відображення

$$\omega: A_{i_1} \times A_{i_2} \times \ldots \times A_{i_n} \to A_j.$$

Якщо $n = 0$, то тип s має вигляд $s = (j)$ і тоді ω є *нульарною операцією*, яка виділяє елемент s у множині A_j.

Визначення 55. **Багатоосновною алгеброю** *називається пара* $G = (U, \Omega)$, *де* U — *деякий комплект, а* Ω — *сигнатура операцій, які визначені на комплекті* U.

На багатоосновні алгебри легко переносяться розглянуті вище визначення і конструкції теорії універсальних алгебр.

Нехай $U = \{A_i, i \in I\}$, $U' = \{A_i', i \in I\}$ — два комплекти. Комплект U' включається в комплект U ($U' \subseteq U$), якщо $A_i' \subseteq A_i$ для довільного i із I. Можна говорити також і про теоретикомножинні операції над комплектами, які зводяться до операцій над відповідними компонентами множин з одним і тим самим ім'ям. Наприклад, $U \cup U' = \{A_i \cup A_i', i \in I\}$, де $U = \{A_i, i \in I\}$ і $U' = \{A_i', i \in I\}$.

Якщо $U' \subseteq U$, то алгебра $G' = (U', \Omega)$ називається **підалгеброю** алгебри $G = (U, \Omega)$, якщо комплект U' замкнений відносно довільної операції із Ω. Багатоосновні алгебри $G = (U, \Omega)$ і $G' = (U', \Omega')$ називаються алгебрами одного й того самого типу, якщо вони мають одну й ту саму сигнатуру, тобто $\Omega = \Omega'$, і комплекти з одними й тими самими іменами, тобто $U = \{A_i, i \in I\}$ і $U' = \{A'_i, i \in I\}$.

Відображення φ алгебри $G = (U, \Omega)$ в алгебру $G' = (U', \Omega)$ одного з нею типу, називається **гомоморфізмом,** якщо для довільної операції $\omega \in \Omega$ типу $s = (i_1, i_2, \ldots, i_n : j)$ і довільних елементів $a_i \in A_{i_1}, \ldots, a_i \in A_{i_n}$ має місце рівність

$$\varphi(\omega(a_1, a_2, \ldots, a_n)) = \omega(\varphi(a_1), \varphi(a_2), \ldots, \varphi(a_n)),$$

де $\varphi: A_i \to A_i'$ для всіх i з I.

Якщо для довільного i з I відображення φ взаємно однозначне, то воно називається **ізоморфізмом** алгебр G і G'.

Якщо $U' = \{A_i', i \in I\}$ — довільний підкомплект комплекту U, то перетин усіх підалгебр алгебри $G = (U, \Omega)$ з непустими A_i, $i \in I$, які включають U', є підалгеброю алгебри G, що породжена комплектом U'. Якщо U' породжує U, то позначають це через $\{U'\} = U$.

У випадку багатоосновних алгебр, як і у випадку одноосновних алгебр, визначають множину термів. Нехай Ω — деяка множина багатосортних операцій, а $X = \{X_i, i \in I\}$ — деякий комплект.

Визначення 56. *Множина многосортних термів визначається індуктивно таким чином: термами є*

а) усі елементи із X_i і всі нульарні символи операцій типу (i);

б) якщо $\omega \in \Omega$ типу $s = (i, j, \ldots, k; l)$ і $t_1, t_2, \ldots, t_n$ — терми типу $i, j, \ldots, k$ відповідно, то $\omega(t_1, t_2, \ldots, t_n)$ — терм типу l;

в) других термів немає.

Багатоосновну алгебру многосортних термів називають **абсолютно вільною багатоосновною алгеброю термів.**

2.5.1. Алгебра алгоритмів Глушкова

Важливим прикладом багатоосновної алгебри є алгебра алгоритмів, яку запропонував В. М. Глушков у 1965 р. Перейдемо до розгляду основних понять алгебри алгоритмів.

Нехай B — деяка множина, яку будемо називати *інформаційним середовищем*. Пов'яжемо з множиною B дві алгебри ОП і УМ, які будемо називати відповідно **алгебрами операторів** і **умов.** Елементами алгебри ОП є (часткові) перетворення множини B, які називають *операторами*, а елементами алгебри УМ — (часткові) предикати, що визначені на множині B і які називають *умовами*.

Крім звичайної операції множення (суперпозиції) відображень в алгебрі ОП визначаються ще дві операції: **α-диз'юнкцію** і **α-ітерацію операторів**, що пов'язують алгебри ОП і УМ.

Результатом α-диз'юнкції ${}_\alpha P \vee Q$ двох операторів P і Q буде такий оператор R, який для довільного $b \in B$ збігається з $P(b)$, коли $\alpha(b) = 1$, або з $Q(b)$, коли $\alpha(b) = 0$, або невизначений, коли $\alpha(b)$ не визначено.

Результатом α-ітерації $\{{}_\alpha P\}$ оператора P є оператор R такий, що $\forall b \in B$ $R(b)$ дорівнює першому із станів пам'яті в рядку b, $P(b)$, $P^2(b), \ldots, P^n(b), \ldots$, для якого виконується умова α. Якщо такого стану немає, то результат застосування оператора R до стану b вважається невизначеним.

Операціями в алгебрі умов УМ служать операції диз'юнкції ($\vee$), кон'юнкції ($\wedge$), заперечення ($\neg$) і (ліве) множення умови на оператори з ОП. Для перших трьох операцій пояснення вимагають лише ті ситуації, коли разом зі звичайними значеннями умов — 0 і 1 — трапляються невизначені значення *. Правила виконання операцій в цьому випадку задаються такими співвідношеннями:

$$1 \vee * = * \vee 1 = 1; \quad 0 \vee * = * \vee 0 = *;$$
$$1 \wedge * = * \wedge 1 = *; \quad 0 \wedge * = * \wedge 0 = 0;$$
$$* \vee * = *; \quad * \wedge * = *; \quad \neg * = *.$$

Зауважимо, що відповідно до наведених співвідношень вираз $* \vee \neg *$ дорівнює *, а не 1, оскільки $* \vee \neg * = * \vee * = *$.

Нехай $\alpha \in$ УМ, $P \in$ ОП — довільні умова і оператор, $b \in B$. Тоді $P\alpha$ є умовою β такою, що $\beta(b) = 1 \Leftrightarrow \alpha(P(b)) = 1$, $\beta(b) = 0 \Leftrightarrow \alpha(P(b)) = 0$ і $\beta(b) = *$, якщо $P(b)$ або $\alpha(P(b))$ невизначене.

Якщо в алгебрі операторів і умов зафіксувати деякі системи твірних (базиси), то елементи кожної з цих алгебр можна задавати виразами, що складаються з твірних і символів операцій системи $Z =$ = (ОП, УМ), які описувалися вище. Такі вирази називаються **регулярними**, а сама система $Z =$ (УМ, ОП) — **алгеброю алгоритмів**.

Зокрема, алгебра алгоритмів дозволяє представляти програми у вигляді виразів алгебри алгоритмів і проводити над програмами формальні перетворення. Ці перетворення ґрунтуються, насамперед, на **тотожних** і **квазітотожних співвідношеннях** алгебри алгоритмів і деяких інших співвідношеннях.

Визначення 57. *Тотожні співвідношення алгебри алгоритмів — це співвідношення, що виконуються для всіх операторів і умов незалежно від станів інформаційного середовища B.*

Квазітотожностями називаються співвідношення, що виконуються при деяких вимогах, накладених на умови і оператори, які входять до їх запису, незалежно від стану інформаційного середовища B.

Приклади тотожностей алгебри алгоритмів

$({}_{\alpha}P \vee Q) = ({}_{\neg\alpha}Q \vee P)$;
$P({}_{\alpha}Q \vee R) = ({}_{P\alpha}PQ \vee PR)$;
$({}_{\alpha}QP \vee RP) = ({}_{\alpha}Q \vee R)P$;
$({}_{\alpha}({}_{\beta}P \vee Q) \vee R) = ({}_{\alpha\&\neg\beta}P \vee ({}_{\alpha\&\neg\beta}Q \vee R))$;
$\{{}_{\alpha}e\} = e$, $\{{}_{\alpha}\{{}_{\alpha}P\}\} = \{{}_{\alpha}P\}$.

$P\alpha = \alpha \Rightarrow ({}_\alpha PQ \vee PR) = P({}_\alpha Q \vee R)$;

$P\alpha = \alpha,\ PQ = QP \Rightarrow P\{{}_\alpha Q\} = \{{}_\alpha Q\}P$;

$P\alpha = \alpha,\ PQ = QP,\ P^2 = P \Rightarrow \{{}_\alpha PQ\} = \{{}_\alpha QP\} = ({}_\alpha e \vee P)\{{}_\alpha Q\}$.

2.6. Повні ґратки і напівкільця

У цьому короткому підрозділі розглядаються алгебраїчні ґратки, які не є універсальними алгебрами в тому розумінні, в якому вони визначалися вище. Багато універсальних алгебр мають таку властивість, що перетин, об'єднання або сума визначені в них не тільки для двох, і тому в силу закону асоціативності для довільного скінченного числа елементів, але й для нескінченного (зліченного) числа елементів. Такі алгебри не є універсальними, але вони існують і мають застосування. Розглянемо дві з таких алгебр — повні ґратки і напівкільця.

2.6.1. Повні ґратки

Визначення 58. *Нехай A — частково упорядкована множина відношенням часткового порядку $\leq$.*

Множина A називається **повною ґраткою**, *якщо для довільної непустої підмножини B множини A у множині A існують такі елементи c і d, для яких виконуються умови:*

(а) для всіх елементів $a \in B$ $c \leq a$ і якщо існує деякий елемент c', такий, що $c' \leq a$, то $c' \leq c$,

(б) для всіх елементів $a \in B$ $d \geq a$, і якщо існує деякий елемент d', такий що $d' \geq a$, то $d' \geq d$.

Однозначно визначені елементи c і d називаються відповідно *перетином* і *об'єднанням* елементів підмножини B і їх записують $c = \cap B$ (або $c = \bigcap_{a \in B} a$ або $c = \bigcap_{i \in I} a_i$, якщо a_i пробігає всі елементи множини B), $d = \cup B$ (або $d = \bigcup_{a \in B} a$ або $d = \bigcup_{i \in I} a_i$).

Очевидно, що нескінченна повна ґратка не є універсальною алгеброю, але зрозуміло, що вона буде ґраткою. Крім того, очевидно також, що повна ґратка має нуль і одиницю — це будуть відповідно елементи $\cap A$ і $\cup A$.

Теорема 56. *Якщо частково упорядкована множина A має одиницю і в ній існує перетин для довільних непустих підмножин, то A є повною ґраткою [34, 17, 2].*

Доведення. Необхідно довести, що довільна непуста підмножина B множини A має об'єднання.

Нехай C — множина всіх таких елементів b із A, що $b \geq a$ для всіх a із B. Множина C непуста, оскільки $1 \in C$, а отже, існує $d = \cap C$. Покажемо, що $d = \cup B$.

Дійсно, якщо $a \in B$, то для довільного b із C $a \leq b$ і, отже, $a \leq d$.

З іншого боку, якщо елемент $s \in A$ такий, що для довільного $a \in B$ $s \geq a$, *то* $s \in C$ і тоді $d \leq s$. ■

Нехай A і A' — частково упорядковані множини відношенням часткового порядку $\leq$ і φ: $A \to A'$ — деяке відображення множини A у множину A'.

Визначення 59. *Відображення* φ *називається* **ізотонним,** *якщо із того, що* $a \leq b$, *випливає* $\varphi(a) \leq \varphi(b)$. *Елемент* $a \in A$ *називається* **нерухомою точкою ізотонного відображення** φ *множини* A *в себе, якщо* $\varphi(a) = a$.

Теорема 57 (*Теорема про нерухому точку*). *Якщо* φ — *ізотопне відображення повної ґратки* A *в себе, тобто* φ: $A \to A$, *то* $\varphi(a) = a$ *для деякого* a *із* A.

Доведення. Нехай B — множина всіх таких елементів s із A, що $s \leq \varphi(s)$. Зрозуміло, що $0 \in B$, оскільки A — повна ґратка і, отже, $B \neq \varnothing$. Тоді існує $a = \cup B$. Причому для довільного $s \in B$ маємо $\varphi(a) \geq \varphi(s) \geq s$ у силу ізотонності відображення φ. Звідси випливає, що $\varphi(a) \geq a$. Але тоді $\varphi(\varphi(a)) \geq \varphi(a)$, звідки маємо, що $\varphi(a) \in B$ і, отже, $\varphi(a) \leq a$. Таким чином, $a \leq \varphi(a) \leq a$, тобто $\varphi(a) = a$, що й потрібно було довести.

На жаль, теорема, обернена до теореми про нерухому точку, не має місця, як показує такий простий приклад.

Нехай $A = \{a, b, c\}$ — триелементна частково упорядкована множина, де $a \leq b$ і $c \leq b$. Очевидно, що дана множина не є повною ґраткою, але разом з тим довільне ізотонне відображення A в A має нерухому точку.

Приклади повних ґраток

1. Ґратка $B(U)$ — усіх підмножин довільної множини U є повною ґраткою.

2. Ґратка всіх підалгебр довільної алгебри G є повною ґраткою. ♠

2.6.2. Замкнуті напівкільця

Визначення 60. *Нехай на деякій непустій множині A задано сигнатуру операцій Ω, яка включає дві бінарні операції додавання $(+)$ і множення $(\cdot)$, а також дві нульарні операції 0 і 1. Якщо ці операції задовольняють властивостям:*

1) $x + (y + z) = (x + y) = z$ — *асоціативність*;
2) $x + y = y + x$ — *комутативність*;
3) $x + x = x$ — *ідемпотентність*;
4) $x + 0 = 0 + x = x$
5) $x \cdot (y \cdot z) = (x \cdot y) \cdot z$ — *асоціативність*;
6) $x \cdot 0 = 0 \cdot x = 0$
7) $x \cdot 1 = 1 \cdot x = x$
8) $x \cdot (y + z) = x \cdot y + x \cdot z$ — *дистрибутивність*;
   $(y + z) \cdot x = y \cdot x + z \cdot x$
9) *зліченна сума вигляду* $a_0 + a_1 + a_2 + \ldots$ *існує, єдина і належить множині* A;
10) *операція множення дистрибутивна відносно нескінченних зліченних сум, тобто*

$$\sum_i a_i \cdot \sum_j b_j = \sum_{i,j} a_i \cdot b_j = a_1 \cdot b_1 + a_1 \cdot b_2 + \ldots$$

$$\ldots + a_2 \cdot b_1 + a_2 \cdot b_2 + \ldots + a_3 \cdot b_1 + a_3 \cdot b_2 + \ldots,$$

то така алгебра називається **напівкільцем.**

Ця алгебра застосовується при обчисленні значень досить важливої унарної операції **замикання**, яку в напівкільцях позначають $*$. Результатом застосування цієї операції до елемента a є елемент вигляду

$$a^* = \sum_{i=0}^{\infty} a^i,$$

де $a^0 = 1$, $a^1 = a$, $a^i = a \cdot a^{i-1}$.

Приклади. Неважко впевнитися в тому, що наведені нижче алгебри є напівкільцями.

а) $G = (\{0, 1\}, \{\vee, \wedge, 0, 1\})$, де операції $\vee$ і $\wedge$ задаються таблицями

$\vee$	0	1
0	0	1
1	1	1

$\wedge$	0	1
0	0	0
1	0	1.

Властивості 1) — 8) легко перевірити. Що ж стосується властивостей 9) і 10), то слід зауважити, що

$$\bigvee_{i=0}^{\infty} a_i = 0 \Leftrightarrow (\forall i \in N) a_i = 0 \text{ і } \bigvee_{i=0}^{\infty} a_i = 1 \Leftrightarrow (\exists i \in N) a_i = 1.$$

б) $G_2 = (D^+ \cup \{0\}, \{\min, +, \infty, 0\})$, де $D^+ = \{x \in D \mid x > 0\}$, а $\min(a, b)$ — це найменше з чисел a і b. Роль мультиплікативної одиниці відносно операції «+» відіграє 0, а роль адитивної одиниці відносно операції min — ∞.

в) Нехай $F(X)$ — вільний моноїд слів у деякому алфавіті X і e — пусте слово. Тоді $G_3 = (F(X), \{\cup, ;\ \varnothing, \cdot 0, \{e\}\})$ — напівкільце, де $\cup$ — об'єднання множин слів, а операція множення множин L і L' визначається так:

$$L \cdot L' = \{pq \mid p \in L, q \in L'\},$$

де pq — слово, отримане в результаті операції конкатенації.

Відносно законів 9) і 10) необхідно зауважити, що

$$p \in \bigcup_{i=0}^{\infty} A_i \Leftrightarrow p \in A_i$$

для деякого i із N.

г) Алгебра $G_4 = (\{0, 1\}, \{+, \cdot, 0, 1\})$, де операції + і · задаються таблицями

+	0	1
0	0	1
1	1	0

·	0	1
0	0	0
1	0	1.

не є замкнутим напівкільцем, оскільки операція «+» не задовольняє закон ідемпотентності.

Для такої алгебри операція замикання не має сенсу, тому що

$$1 + 1 + 1 + 1 + \ldots + 1 + \ldots = (1 + 1) + (1 + 1) + \ldots + (1 + 1) + \ldots = 0,$$

і водночас

$$1 + 1 + 1 + 1 + \ldots + 1 + \ldots = 1 + (1 + 1) + (1 + 1) + \ldots + (1 + 1) + \ldots = 1.$$

А це означає, що нескінченна зліченна сума невизначена. ♠

Якщо $G = (A, \Omega)$ — замкнуте напівкільце і $a \in A$, то

$$a^* = 1 + a + a^2 + \ldots .$$

Властивість 9) гарантує, що $a^* \in A$, а із 9) і 10) випливає, що $0^* = 1^* = 1$, а також що $a^* = 1 + a \cdot a^*$.

Наприклад, у напівкільцях, що були наведені вище як приклади, маємо:

а) $a^* = 1$, для $a = 1$ і для $a = 0$;

б) $a^* = 0$, для довільного a із A;

в) $L^* = \{e\} \cup L \cup L^2 \cup L^3 \cup \ldots = \{e\} \cup \{p_1, p_2 \ldots p_k \mid p_1, p_2, \ldots p_k \in L, k \geq 1\}$ для всіх L із $F(X)$. Зокрема, якщо $L = \{a, b\}$, то

$$L^* = \{e\} \cup \{a, b\} \cup \{aa, ab, ba, bb\} \cup \{aaa, aab, abb, aba, \ldots\} \ldots .$$

2.7. Контрольні питання, задачі і вправи

Контрольні питання

1. Яке відношення називається відношенням конгруентності?
2. Дайте визначення підалгебри алгебри?
3. Яке найбільше число підалгебр може мати алгебра, яка складається з чотирьох елементів?
4. Що таке абсолютно вільна алгебра?
5. Чи буде скінченною алгебра термів, у якої $X = \{x, y, a, b\}$ і $F = \{+, -, *, /\}$?
6. Дайте визначення гомоморфізму (ізоморфізму) векторних просторів.
7. Чи буде довільна група напівгрупою?
8. Яка різниця між кільцем і полем?
9. Побудуйте ізоморфізм між напівгрупами $G = \{e, a, a^2, a^3, \ldots\}$ і N, де N — множина натуральних чисел з основною операцією додавання, а основною операцією на G є множення ($a^k a^l = a^{k+l}$).
10. Наведіть приклад кільця з дільниками нуля.
11. Поясніть, чому напівкільце не є універсальною алгеброю.
12. Наведіть приклад повної ґратки.

Задачі і вправи

1. Доведіть твердження 2.
2. Доведіть, що напівгрепа, в якій виконується закон скорочення, має єдиний одиничний елемент.
3. Перевірити виконання закону асоціативності для алгебр, які задані такими таблицями Келі:

*	e	f	g	a
e	e	e	e	e
f	f	f	f	f
g	g	g	g	g
a	e	e	f	e,

*	e	f	g	a	0
e	e	a	e	a	0
f	0	f	g	0	0
g	g	f	g	f	0
a	0	a	e	0	0
0	0	0	0	0	0

4. Доведіть, що в групі одиничний елемент єдиний, і що довільний елемент групи має єдиний обернений елемент.

5. Доведіть, що гомоморфний образ напівгрупи, групи, кільця є відповідно напівгрупою, групою, кільцем.

6. Чи може група бути ізоморфною своїй власній підгрупі? Навести приклад і відповідне обґрунтування.

7. Чи може вільна група мати абелеву підгрупу? Наведіть приклад.

8. Доведіть, що довільна ненульова підгрупа адитивної циклічної групи сама є циклічною групою.

9. Нехай G — нескінченна група, породжена елементом a, і нехай H — циклічна група другого порядку з елементами e, b, де $b^2 = e$. Довести, що існує гомоморфізм групи G на групу H, а ізоморфізму цих груп не існує.

10. Нехай G — група і a — її фіксований елемент. Якщо x — довільний елемент групи G, то визначимо відображення f: $G \to G$ формулою $f(x) = axa^{-1}$. Довести, що f — автоморфізм групи G.

11. Нехай G — група і f: $G \to G$ таке, що $f(x) = x^2$, де $x \in G$. Чи буде відображення f ізоморфізмом? Чи може бути відображення f гомоморфізмом, і якщо так, то для яких груп?

12. Доведіть, що довільна скінченна група, яка складається із n елементів, ізоморфна симетричній групі підстановок деякої множини M із n елементів.

13. а) Побудуйте групу підстановок, яка ізоморфна циклічній групі четвертого порядку $G = \{e, a, a^2, a^3\}$;

б) Перевірте, чи виконуються такі рівності в цій групі підстановок $f_2^2 = f_3$, $f_3^2 = e$, $f_2^3 = f_4$, $f_2 f_4 = e$, де $f_1 = e$ — тотожна підстановка, а f_2, f_3, f_4 — решта елементів групи підстановок.

14. Сформулюйте і доведіть теорему про гомоморфізми для напівгруп, груп, кілець.

15. Доведіть, що слова, які складають множину елементів вільної групи, задовольняють асоціативному закону.

16. Доведіть, що скінченна напівгрупа з одиницею і законом (лівого або правого) скорочення є групою.

17. Доведіть, що підгрупа H групи G індексу 2 буде нормальним дільником групи G.

18. Довести, що коли порядок групи G дорівнює $2n$, де n — натуральне число більше за 1, і H — її підгрупа n-го порядку, то H — нормальний дільник групи G.

19. Нехай група $G = \{e, g_1, g_2, \ldots\}$ і $x \in G$ — довільний елемент групи. Довести, що множина $S = \{e, xg_1x^{-1}, xg_2x^{-1}, \ldots\}$ включає всі елементи групи G.

20. Нехай R і S — підгрупи групи G. Визначимо добуток двох підгруп R і S таким чином: $R \cdot S = \{rs \mid r \in R \text{ і } s \in S\}$. Довести, що

а) множина $R \cdot S$ буде підгрупою групи G тоді і тільки тоді, коли $R \cdot S = S \cdot R$;

б) якщо одна із підгруп (R чи S) є нормальним дільником групи G, то $R \cdot S = S \cdot R$ буде підгрупою групи G.

21. Довизначіть таблицю додавання GN_5:

+	0	1	2	3	4
0	0	1	2	3	4
1	1	3	4	2	0
2	2	4			
3	3	2			
4	4	0			

22. Поясніть, чому не можна коректно довизначити таблицю додавання

+	0	1	2	3	4	5
0	0	1	2	3	4	5
1	1	0	3	5	2	4
2	2	3				
3	3	5				
4	4	2				
5	4	4				

(щоб упевнитися, що дане визначення некоректне, знайдіть, чому дорівнює 1 + 3).

23. Покажіть, що коли бінарним відношенням R і R_1 відповідають матриці $A(R)$ і $A(R_1)$, то відношенню $R * R_1$ відповідає матриця $A(R) \cdot A(R_1)$. Які корисні властивості можна отримати з цього?

24. Покажіть, що порядок симетричної групи n-го степеня дорівнює $n!$.

25. Доведіть, що ядро *ker* (*h*) гомоморфізму *h* групи *G* в групу *G'* є нормальним дільником групи *G*.

26. Чи буде гомоморфізмом відображення *h* із прикладу гомоморфізму абелевих груп, якщо його задати так:

$$h(n)=\begin{cases} 1, \text{ коли } n \text{ — непарне,} \\ -1, \text{ коли } n \text{ — парне.} \end{cases}$$

27. Наведіть доведення теореми 42.

28. Знайдіть кільце, яке задане наведеною нижче таблицею додавання, де елемент 1 відіграє роль одиниці кільця.

+	0	1	2	3	4	5
0	0	1	2	3	4	5
1	1	3	0	5	2	4
2	2	0				
3	3	5				
4	4	2				
5	5	4				

29. Побудуйте кільце для а) «трійкової» (ліва таблиця) і б) «п'ятіркової» арифметики (права таблиця)

+	0	1	2
0	0	1	2
1	1	2	0
2	2	0	1,

+	0	1	2	3	4
0	0	1	2	3	4
1	1	2	3	4	0
2	2	3	4	0	1
3	3	4	0	1	2
4	4	0	1	2	3

30. Знайти розклад $(a + b)^3$ у некомутативному кільці.

31. Довести, що булеве кільце, яке має більше двох елементів, не може бути областю цілісності. (*Підказка*: розглянути рівняння $x(x-1)=0$).

32. Довести, що довільне скінченне асоціативно-комутативне кільце, яке є областю цілісності, буде полем.

33. Довести, що в комутативному кільці, яке є областю цілісності, не існує ідемпотентних елементів (тобто, елементів x таких, що $x^2 = x$), крім 0 і 1.

34. Дайте означення підалгебри булевої алгебри. Чи буде пара $(\{0, 1\}, \{\vee, \wedge, \neg, 0, 1\})$ підалгеброю булевої алгебри $G = (A, \{\vee, \wedge, \neg, 0, 1\})$?

Нехай $G_1 = (\{0, x, \neg x, 1\}, \{\vee, \wedge, \neg, 0, 1\})$. Чи буде G_1 підалгеброю булевої алгебри $G = (A, \{\vee, \wedge, \neg, 0, 1\})$, якщо $x \in A$?

35. Нехай $G_1 = (A, \{\vee, \wedge, \neg, 0_A, 1_A\})$ і $G_2 = (B, \{\vee, \wedge, \neg, 0_B, 1_B\})$ — булеві алгебри. Алгебра $G = (A \times B, \Omega)$ називається декартовим добутком алгебр G_1 і G_2, якщо

$$(a, b) \vee (a_1, b_1) = ((a \vee a_1), (b \vee b_1)),$$
$$(a, b) \wedge (a_1, b_1) = ((a \wedge a_1), (b \wedge b_1)),$$
$$\neg(a, b) = (\neg a, \neg b),\ 0 = (0_A, 0_B),\ 1 = (1_A, 1_B)$$

Довести, що алгебра G буде булевою алгеброю.

36. З означення булевої алгебри випливає, що булева алгебра є дистрибутивною ґраткою, тобто частково упорядкованою множиною.

Довести, що взаємно однозначна відповідність між елементами двох булевих алгебр буде ізоморфізмом цих алгебр тоді і тільки тоді, коли ця відповідність зберігає частковий порядок.

37. Доведіть, що для довільних елементів деякої ґратки G мають місце такі властивості:

а) із $a \le b$ і $c \le b$ випливає $a \vee c \le b$;

б) із $a \le b$ випливає, що для довільного c із G $a \vee c \le b \vee c$;

в) із $a \le b$ і $a \le c$ випливає $a \le b \wedge c$;

г) із $a \le b$ випливає, що для довільного c із G $a \wedge c \le b \wedge c$;

д) $a \vee (b \wedge c) \le (a \vee b) \wedge (a \vee c)$;

є) $a \wedge (b \vee c) \ge (a \wedge b) \vee (a \wedge c)$;

ж) із $a \le b$ випливає $a \vee (b \wedge c) \le (a \vee b) \wedge c$.

38. Доведіть, що всі можливі бінарні відношення, задані на деякій множині A і упорядковані відносно включення множин, утворюють повну ґратку.

39. Чи будуть повними ґратками відносно включення всі рефлексивні, симетричні, антисиметричні, транзитивні відношення, задані на деякій непустій множині A?

40. Доведіть, що довільний ланцюг є ґраткою.

41. Побудуйте ДНФ і ДДНФ для булевих функцій:

$\neg(x \wedge y) \wedge (x \wedge z) \vee \neg(y \vee z)$; $\neg(\neg x \wedge \neg y) \wedge (x \wedge z) \vee \neg(\neg y \vee z)$;

$(\neg x \wedge y) \wedge (x \wedge z) \vee \neg(y \vee \neg z)$.

42. Довести, що коли в ґратці виконується тотожність $a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c)$, то в цій ґратці виконується тотожність $a \vee (b \wedge c) = (a \vee b) \wedge (a \vee c)$.

43. Довести, що множина натуральних чисел N є повною ґраткою відносно відношення подільності |, тобто $mRn \Leftrightarrow m$ націло ділить n.

3 ОСНОВНІ АЛГОРИТМІЧНІ СИСТЕМИ

Термін «алгоритм» уже вживався нами у попередньому розділі під час розгляду алгебри булевих функцій. У цьому розділі зазначене поняття буде детально проаналізовано.

Алгоритму, як поняттю, вже понад тисячу років. Протягом цього часу це поняття використовувалось тільки математиками, які тлумачили його як послідовність правил розв'язування довільної задачі з того, чи іншого класу або правил обчислення значення функції, яка задана тим чи іншим способом. Таке інтуїтивне тлумачення поняття «алгоритм» до певного часу влаштовувало дослідників і, навіть, на початку XX-го століття панувала думка, що для кожного класу задач повинен існувати (принаймні в принципі) алгоритм розв'язання кожної задачі з цього класу.

Майже тисячу років не виникало потреби в уточненні і формалізації поняття «алгоритм». Але на початку тридцятих років минулого століття у зв'язку з розвитком формальних методів у математиці і, зокрема, аксіоматичного підходу до обґрунтування основ математики, виникла потреба в строгому понятті алгоритму. Ця потреба стала ще нагальнішою, оскільки з'явилися сумніви в необмежених можливостях алгоритмічних методів. Так, у 1931 році австрійський математик Курт Гедель показав, що для звичайної арифметики в аксіоматизації Пеано не існує єдиної системи правил, за допомогою якої можна перевірити істинність довільного твердження цієї арифметики.

Після появи цієї праці К. Геделя було проведено багато спроб сформулювати точно поняття «алгоритм». Проте виявилося, що ці спроби зіштовхувалися з труднощами логічного характеру і у зв'язку з цим спочатку з'явилось інтуїтивне визначення алгоритму, а потім — різного роду уточнення цього інтуїтивного визначення. Різноманітні конкретизації визначення алгоритму в термінах машин чи ітеративних процесів пов'язані з іменами А. М. Тьюрінга, Е. Л. Поста, А. Черча, Дж. фон Неймана, А. А. Маркова, С. К. Кліні, А. Н. Колмогорова, В. А. Успенського. Усі ці формалізації еквівалентні між собою.

Перейдемо до розгляду деяких найвідоміших способів уточнення поняття «алгоритм».

3.1. Поняття алгоритму та алгоритмічної системи

3.1.1. Інтуїтивне поняття алгоритму

Інтуїтивне поняття алгоритму включає в себе декілька загальних рис, які чітко вимальовуються з попередніх прикладів алгоритмів побудови ДНФ і КНФ і часто визнаються характерними для поняття алгоритму.

1) **Алгоритм** — це процес послідовної побудови величин, який відбувається в дискретному часі таким чином, що в початковий момент задається висхідна скінченна множина величин, а в кожний наступний момент система величин знаходиться за цілком визначеним законом (програмою) із системи величин, які були в попередній момент часу **(дискретність алгоритму)**.

2) Система величин, що отримується в якийсь відмінний від початкового момент часу, однозначно визначається системою величин, отриманих у попередні моменти часу **(детермінованість алгоритму)**.

3) Закон отримання подальших систем величин із попередніх має бути простим і локальним **(елементарність кроків алгоритму)**.

4) Якщо спосіб отримання наступної величини із якої-небудь заданої величини не дає результату, то має бути вказано, що треба вважати результатом алгоритму **(направленість алгоритму)**.

5) Початкова система величин може вибиратись із деякої потенціально нескінченної множини **(масовість алгоритму)**.

Поняття алгоритму, деякою мірою визначене пунктами 1)—5), звичайно, не строге: у формулюваннях цих пунктів використовуються терміни «спосіб», «величина», «простий», «локальний», точний зміст яких не встановлено. Надалі це нестроге поняття алгоритму буде називатися **безпосереднім** або **інтуїтивним поняттям алгоритму.**

Користуючись інтуїтивним поняттям алгоритму можна описувати процес розв'язку тієї чи іншої задачі, але за його допомогою не можна упевнитися в тому, що описаний процес являє собою алгоритм. Дійсно, одна справа довести існування алгоритму, а зовсім інша — довести його відсутність. Довести відсутність алгоритму таким шляхом неможливо. Для цього потрібно знати точно, що таке алгоритм. Розв'язок цієї задачі було отримано в середині тридцятих років XX століття у двох формах. Перша ґрунтувалась на понятті рекурсивної функції, а друга — на точно окресленому класі процесів. Обидві форми отримали назву **алгоритмічні системи.**

3.1.2. Обчислювані і частково рекурсивні функції. Теза Черча

Нехай N — множина натуральних чисел і $F = \{f_1, f_2, ..., f_n, ...\}$ — сукупність n-арних функцій на множині N.

Далі ми часто будемо використовувати унарні функції o, s і n-арні функції $I_m^n(x_1, ..., x_n)$, які визначаються таким чином:

- $o(x) = 0$ (нуль-функція),
- $s(x) = x + 1$ (функція «наступний»),
- $I_m^n(x_1, ..., x_n) = x_m$, $1 \le m \le n$ (селекторні функції).

Ці функції будемо називати **найпростішими.** До найпростіших відносять також і n-арну нуль-функцію $o^n(x_1, ..., x_n)$, що дорівнює 0 для всіх $x_1, x_2, ..., x_n \in N$.

Розглянемо $n + 1$ функцію із F — функцію f-арності n і функції $f_1, ..., f_n$ однієї й тієї самої арності m.

Вважають, що m-арну часткову функцію $g(x_1, ..., x_m)$ одержано в результаті **операції суперпозиції** або **підстановки** (S^{n+1}) із функцій f^n, $f_1^m, ..., f_n^m$, якщо для довільних $x_1, ..., x_m \in N$

$$g^m(x_1, ..., x_m) = f^n(f_1^m(x_1, ..., x_m), ..., f_n^m(x_1, ..., x_m)),$$

де деякі із x_i можуть входити у f_j і фіктивно.

Приклади. 1) Для n-арної функції $o^n(x_1, ..., x_n) = 0$ маємо

$$o^n(x_1, ..., x_n) = S^2(o, I_1^n) = o(I_1^n(x_1, ..., x_n)) = o(x_1) = 0.$$

Тобто, n-арну функцію $o^n(x_1, ..., x_n) = 0$ отримують за допомогою операції суперпозиції з найпростіших функцій o і I_1^n.

2) Розглянемо n-арну функцію $f(x_1, ..., x_n) = a$, де a — константа. Тоді

$$a = \underbrace{s(...s}_{a \text{ разів}}(o^n(x_1, ..., x_n))...),$$

тобто константу a отримуємо за допомогою операції суперпозиції з найпростіших функцій o і s. ♠

Нехай задано довільні часткові функції: n-арна функція g і $n + 2$-арна функція h. Вважають, що $n + 1$-арна функція f отримана **операцією примітивної рекурсії** з функцій g, h (**R(g, h)**), якщо для всіх $x_1, ..., x_n, y \in N$ маємо

$$f(x_1, ..., x_n, 0) = g(x_1, ..., x_n),$$
$$f(x_1, ..., x_n, y + 1) = h(x_1, ..., x_n, y, f(x_1, ..., x_n, y)).$$

Визначення 61. *Нехай Z — деяка система часткових функцій. Функція f називається* **примітивно рекурсивною відносно системи функцій** *Z, якщо f можна отримати з функцій системи Z і найпростіших функцій o, s, I_m^n за допомогою скінченного числа операцій підстановки і примітивної рекурсії. Примітивно рекурсивна функція f відносно пустої системи функцій Z називається просто* **примітивно рекурсивною функцією**.

Зауважимо, що операції підстановки і примітивної рекурсії, будучи застосованими до повністю визначених функцій, дають знову повністю визначені функції. Зокрема, усі примітивно рекурсивні функції є повністю визначені.

Приклади. 1) Функцію $f(x, y) = x + y$ можна отримати з функцій $I_1^1(x)$ і $h(x, y, z) = z + 1$ операцією примітивної рекурсії. Дійсно,

- $x + 0 = x = I_1^1(x)$,
- $x + (y + 1) = (x + y) + 1 = s(x + y) = h(x, y, z) = z + 1$.

2) Функція $x - y$ у множині N часткова. Для того щоб зробити цю функцію повністю визначеною на множині N, розглядають обмежену різницю

$$x \div y = \begin{cases} x - y, \text{ якщо } x \geq y, \\ 0, \text{ якщо } x < y. \end{cases}$$

Безпосередньо з визначення цієї функції випливають такі властивості

- $x \div 0 = x$,
- $x \div (y + 1) = (x \div y) \div 1$,
- $(x \div y) \div z = x \div (y + z)$.

Функція $x \div 1$ примітивно рекурсивна, оскільки

- $0 \div 1 = 0 = o(x)$,
- $(x + 1) \div 1 = x = I_1^2(x, y)$,

тобто її можна отримати з найпростіших функцій o і I_1^2 операцією примітивної рекурсії.

З примітивної рекурсивності функції $x \div 1$ і наведених властивостей операцій обмеженої різниці маємо

- $x \div 0 = x = I_1^1(x)$,
- $x \div (y + 1) = (x \div y) \div 1$,

тобто функцію $x \div y$ можна одержати за допомогою операції примітивної рекурсії із функцій $I_1^1(x)$ і $h(x, y, z) = z \div 1$.

Звідси також випливає, що

$$|x - y| = (x \div y) + (y \div x)$$

теж примітивно рекурсивна функція.

3) Функцію $f(x, y) = x \cdot y$ можна отримати з функцій $o(x)$ і $h(x, y, z) = z + x$ операцією примітивної рекурсії. Дійсно,

- $x \cdot 0 = 0 = o(x)$,
- $x \cdot (y + 1) = (x \cdot y) + x$,

оскільки $x + y$ — примітивно рекурсивна функція.

4) Аналогічно попередньому, функцію x^n можна отримати з функцій $g(x) = 1$ і $h(x, y, z) = z \cdot x$ операцією примітивної рекурсії. Дійсно,

- $x^0 = 1 = s(o(x))$,
- $x^{n+1} = x^n \cdot x$. ♠

Розглянемо ще деякі властивості примітивно рекурсивних функцій, які знадобляться надалі.

Теорема 58. *Нехай n-арна часткова функція g примітивно рекурсивна (відносно системи часткових функцій Z). Тоді n-арні функції f, визначені за допомогою рівнянь*

$$f(x_1, x_2, \ldots, x_n) = \sum_{i=0}^{x_n} g(x_1, x_2, \ldots, x_{n-1}, i),$$

$$f(x_1, x_2, \ldots, x_n) = \prod_{i=0}^{x_n} g(x_1, x_2, \ldots, x_{n-1}, i),$$

також примітивно рекурсивні (відносно Z).

Доведення. Із заданих в умові рівнянь випливає, що

$$f(x_1, \ldots, x_{n-1}, 0) = g(x_1, \ldots, x_{n-1}, 0),$$
$$f(x_1, \ldots, x_{n-1}, y+1) = f(x_1, \ldots, x_{n-1}, y) + g(x_1, \ldots, x_{n-1}, y+1),$$
$$f(x_1, \ldots, x_{n-1}, y+1) = f(x_1, \ldots, x_{n-1}, y) \cdot g(x_1, \ldots, x_{n-1}, y+1).$$

Отже, функції f будуються за допомогою операцій примітивної рекурсії з примітивно рекурсивних функцій $f(x_1, x_2, \ldots, x_{n-1}, 0)$ і

$$h(x_1, x_2, \ldots, x_{n-1}, y, z) = z + g(x_1, x_2, \ldots, x_{n-1}, y+1),$$
$$h(x_1, x_2, \ldots, x_{n-1}, y, z) = z \cdot g(x_1, x_2, \ldots, x_{n-1}, y+1),$$

і тому ці функції примітивно рекурсивні.

Розглянемо ще один клас функцій, які часто використовуються. Нехай задано деякі функції $f_i(x_1, \ldots, x_n)$, $i = 1, 2, \ldots, k+1$, і вказані деякі умови (предикати) $u_j(x_1, \ldots, x_n)$, $j = 1, 2, \ldots, k$, які можуть бути для довільної n-ки чисел $x_1, \ldots, x_n$ істинними або хибними.

Припустимо, що ні для якої n-ки чисел ніякі дві умови не можуть бути хибними одночасно (такі умови називаються **взаємно виключаючими**). Тоді функція $f(x_1, \ldots, x_n)$, задана схемою

$$f(x_1,\ldots,x_n)=\begin{cases} f_1(x_1,\ldots,x_n), & \text{якщо} \quad u_1(x_1,\ldots,x_n) = 0,\\ \ldots\ldots\ldots\ldots & \ldots\ldots\ldots\ldots \quad \ldots\ldots\ldots\ldots \quad \ldots \ldots\\ f_k(x_1,\ldots,x_n), & \text{якщо} \quad u_k(x_1,\ldots,x_n) = 0,\\ f_{k+1}(x_1,\ldots,x_n) & \text{для решти} \quad x_1,\ldots,x_n, \end{cases}$$

називається **кусково заданою.**

Теорема 59. *Нехай задані n-арні примітивно рекурсивні (відносно Z) функції* $f_1,\ldots, f_{k+1}$, $u_1,\ldots, u_k$, *де* u_k *— взаємно виключаючі умови,* $i = 1, 2,\ldots, k$. *Тоді кусково задана функція*

$$f(x_1,\ldots,x_n)=\begin{cases} f_1(x_1,\ldots,x_n), & \text{якщо} \quad u_1(x_1,\ldots,x_n) = 0,\\ \ldots\ldots\ldots\ldots & \ldots\ldots\ldots\ldots \quad \ldots\ldots\ldots\ldots \quad \ldots \ldots\\ f_k(x_1,\ldots,x_n), & \text{якщо} \quad u_k(x_1,\ldots,x_n) = 0,\\ f_{k+1}(x_1,\ldots,x_n) & \text{для решти} \quad x_1,\ldots,x_n \end{cases}$$

примітивно рекурсивна (відносно Z) функція.

Дійсно, дану функцію можна представити у вигляді

$$f = f_1\cdot(1-u_1)+\ldots+f_k\cdot(1-u_k)+f_{k+1}\cdot(u_1\cdot u_2\cdot\ldots\cdot u_k).$$

В силу примітивної рекурсивності функцій обмеженої різниці, додавання і множення, одержуємо примітивну рекурсивність функції f.

Зауважимо, що умови u_i в теоремі 59 мають вигляд $u_i = 0$. Оскільки умови $p_i = p_j$, $p_i \le p_j$, $p_i < p_j$ еквівалентні умовам

$$|p_i - p_j| = 0,\ p_i \div p_j = 0,\ 1 \div (p_j \div p_i) = 0,$$

то теорема 59 залишається справедливою і в цих випадках, якщо p_i, p_j — примітивно рекурсивні функції.

Покажемо тепер, що частка від ділення x на y, тобто функція $[x/y]$, і остача від ділення x на y, тобто функція rest (x, y), — примітивно рекурсивні функції. Для цього необхідно довизначити дані функції до повністю визначених. Припустимо для всіх $x \in N$

$$[x/0] = x, \quad \text{rest}\,(x, 0) = x.$$

Очевидно, що визначені таким чином функції зв'язані тотожністю

$$\text{rest}\,(x, y) = x \div (y \cdot [x/y])$$

і отже, з примітивної рекурсивності функції $[x/y]$ випливає примітивна рекурсивність функції rest (x, y).

За визначенням, при $y > 0$ $[x/y] = n$ задовольняє співвідношенню $n \cdot y \le x < (n+1)\cdot y$. Звідси випливає, що n дорівнює кількості нулів у послідовності

$$1 \cdot y \div x,\ 2 \cdot y \div x,\ \ldots,\ n \cdot y \div x,\ \ldots,\ x \cdot y \div x.$$

Тому при $y > 0$ справедлива формула

$$[x/y] = \sum_{i=1}^{x} (1 \div (i \cdot y \div x)).$$

Безпосередньою перевіркою можна впевнитися, що дана формула справедлива і при $y = 0$. У силу примітивної рекурсивності функції $\div$ і теореми 59 отримуємо, що функція $[x/y]$, а разом з нею і функція rest (x, y) примітивно рекурсивні.

Нехай f — n-арна часткова функція із F $(n \geq 1)$. Зафіксуємо перші $n - 1$ аргументів функції f і розглянемо рівняння

$$f(x_1,..., x_{n-1}, y) = x_n$$

відносно змінної y. Припустимо, що є деяка «механічна» процедура обчислення значень функції f, причому значення f невизначено тоді і тільки тоді, коли цей «механізм» працює нескінченно, не видаючи ніякого результату.

Щоб знайти розв'язок y даного рівняння будемо послідовно обчислювати за допомогою нашого «механізму» значення

$$f(x_1,..., x_{n-1}, y)$$

для $y = 0, 1, 2,\ldots$. Найменше число a, для якого має місце

$$f(x_1, \ldots, x_{n-1}, a)\, x_n,$$

позначимо

$$\mu_y(f(x_1,..., x_{n-1}, y) = x_n).$$

Зауважимо, що $\mu_y(f(x_1,..., x_{n-1}, y) = x_n)$ — n-арна часткова функція. Цю функцію позначають M_f і говорять, що вона отримана **операцією мінімізації** часткової функції f.

Розглядаючи введені операції S^i R, M, ми повинні обґрунтувати цю назву, тобто, чи насправді вони будуть операціями в нашому розумінні. Якщо відносно операцій підстановки і мінімізації це цілком очевидно, то для операції рекурсії потрібні деякі пояснення: чи для довільних функцій g і h існує функція f, і чи буде вона єдиною?

Оскільки g і h — функції над множиною N, то відповідь на обидва запитання ствердна. Якщо функція f існує, то з визначення операції примітивної рекурсії отримуємо

$$f(x_1,..., x_n, 0) = g(x_1,..., x_n),$$
$$f(x_1,..., x_n, 1) = h(x_1,..., x_n, 0, g(x_1,..., x_n)),$$
$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$
$$f(x_1,..., x_n, y+1) = h(x_1,..., x_n, y, f(x_1,..., x_n, y)),$$

і тому f визначена однозначно.

***Визначення 62* (Основне визначення).** *Часткова функція f називається* **частковою рекурсивною відносно системи часткових функцій** *Z, якщо f може бути отримана з функцій системи Z і найпростіших функцій o, s, I_m^n за допомогою застосування скінченного числа операцій підстановки, примітивної рекурсії і мінімізації.*
Часткова рекурсивна функція f відносно пустої системи функцій Z називається просто **частковою рекурсивною функцією.**

За визначенням універсальної алгебри пара $G = (F, \{R, M, S^2, \ldots\})$, де F — множина всіх часткових функцій на N з довільним числом аргументів, є частковою універсальною алгеброю. Сукупність усіх часткових рекурсивних функцій відносно якої-небудь системи функцій Z є підалгеброю алгебри G, породженою множиною функцій $Z \cup \{o, s, I_m^n\}$, $m, n = 1, 2, \ldots$. Аналогічне можна сказати й про часткові рекурсивні функції. Алгебра $G = (F, \{R, S^2, S^3, \ldots\})$ є частковою алгеброю. Сукупність усіх часткових примітивно рекурсивних функцій відносно якої-небудь системи функцій $Z \subseteq F$ є підалгеброю алгебри G, породженої множиною

$$Z \cup \{o, s, I_m^n\}, m, n = 1, 2, \ldots .$$

Важливість розглянутих класів функцій полягає у їх зв'язку з класом обчислюваних функцій. Це формулюється у вигляді таких тез.

Теза Черча. *Клас алгоритмічно, або машинно, обчислюваних часткових числових функцій збігається з класом усіх часткових рекурсивних функцій.*

Більш загальною є

Теза Тьюрінга. *Клас функцій, які алгоритмічно обчислюються відносно якої-небудь системи функцій Z, збігається з класом частково рекурсивних функцій відносно системи Z* [26, 28].

Оскільки поняття обчислюваної функції не має точного визначення, то довести ці тези неможливо, але завдяки їм стало можливим надати необхідну точність формулюванням алгоритмічних проблем.

3.1.3. Алгоритмічні проблеми

Під алгоритмічною проблемою будемо розуміти обчислюваність деяких певним чином побудованих функцій. Відповідно до тез Черча-Тьюрінга питання про обчислюваність функції рівнозначне питанню про її рекурсивність. Поняття рекурсивної функції строге і деколи за допомогою звичайної математичної техніки можна безпо-

середньо довести, що функція, котра вирішує проблему, не може бути рекурсивною. Якщо функція, що вирішує дану алгоритмічну проблему, не є рекурсивною, то проблема називається **(алгоритмічно)** нерозв'язуваною. Коли ж функція рекурсивна, то проблема називається **(алгоритмічно)** розв'язуваною.

Довільний алгоритм, як уже зазначалося, являє собою спосіб розв'язку деякої **масової проблеми**, тобто проблеми переробки (обчислення) не одного вхідного слова (одних вхідних даних), а цілої множини вхідних слів (даних) у відповідні їм вихідні слова. Детальний аналіз задач показує, що існують такі класи задач, для розв'язку яких немає і не може бути єдиного універсального способу, тобто ці задачі відносяться до алгоритмічно нерозв'язуваних проблем. Але алгоритмічна нерозв'язуваність проблеми розв'язку задач того чи іншого класу зовсім не означає неможливість розв'язку довільної конкретної задачі із цього класу. Ідеться про неможливість розв'язку всіх задач даного класу одним і тим самим способом.

Для кращого розуміння поняття алгоритмічно розв'язуваної і алгоритмічно нерозв'язуваної проблем розглянемо приклади.

Проблема тотожності в елементарній алгебрі. Для простоти обмежимося випадком, коли вирази будуються з раціональних чисел і літер за допомогою дій додавання, віднімання і множення. Зі шкільного курсу математики добре відомий такий прийом розв'язку вказаної проблеми: використовуючи дистрибутивний закон для множення, розкривають дужки в лівій і правій частинах даної тотожності й виконують зведення подібних членів. Після цього як ліва, так і права частини тотожності перетворюються в поліноми. Тотожність буде справедливою, якщо ці поліноми тотожно рівні між собою. Іншими словами, справедливість тотожності означає, що після перенесення всіх членів перетвореної таким чином тотожності в одну частину ці члени взаємно знищуються, у результаті тотожність перетворюється на тривіальну тотожність $0 = 0$.

Отже, проблема тотожності в елементарній алгебрі алгоритмічно розв'язувана — існує єдиний конструктивний прийом, який дає можливість за скінченне число кроків з'ясувати, є чи ні довільне задане співвідношення тотожністю.

Можна побудувати приклади таких алгебраїчних систем, у яких проблема тотожності нерозв'язувана. До таких алгебраїчних систем належать напівгрупи і групи, задані за допомогою системи твірних та співвідношень, які їх визначають. Уперше такі приклади для напівгруп були знайдені Е. Постом [26, 28, 31], а для груп — П. С. Новиковим [29]. Оскільки ці проблеми формулюються майже однаково як для напівгруп, так і для груп, розглянемо лише одну з них.

Проблема тотожності для напівгруп. Нехай $F(X)$ — вільна напівгрупа, де $X = \{x_1, x_2, ..., x_n\}$, операцією множення в якій є конкатенація, а пусте слово e відіграє роль одиниці. У цій напівгрупі можна ввести довільну множину співвідношень S, які являють собою формальні рівності між двома неоднаковими словами:

$$p_i = q_i \ (i = 1, 2, ...).$$

Два слова в напівгрупі $F(X)$ тотожні між собою відносно заданої системи співвідношень S, якщо одне з них може бути одержане з другого в результаті довільного числа підстановок у друге слово правих частин співвідношень із S замість лівих і, навпаки, лівих замість правих. Наприклад, нехай $X = \{x, y\}$, а $S = (xy = yx)$. Тоді слова $p = xxy$ і $q = yxx$ тотожні між собою, оскільки перше слово може бути одержане з другого в результаті двох підстановок описаного вище типу: $q = yxx \to xyx \to xxy = p$. У разі підстановок у зворотному порядку, слово q можна одержати із слова p, що дає можливість перетворення як слова q в слово p, так і слова p в слово q.

Проблема тотожності слів для напівгруп формулюється так. Нехай у вільній напівгрупі $F(X)$ зі скінченним числом твірних задано скінченну систему співвідношень S. Необхідно знайти єдиний конструктивний спосіб, який дозволяє за скінченне число кроків з'ясувати, є чи ні довільні два слова p і q із $F(X)$ тотожними відносно системи співвідношень S.

Для деяких систем співвідношень S сформульована проблема розв'язувана, але, як показали А. Марков і Е. Пост, існують і такі системи співвідношень S, для яких проблема тотожності слів напівгруп алгоритмічно нерозв'язувана.

3.2. Машини Поста

Як уже зазначалося на початку даного розділу, рекурсивні функції — не єдина алгоритмічна система. Існує багато інших еквівалентних їй алгоритмічних систем: система нормальних алгорифмів Маркова, система Поста, система Тьюрінга, система Колмогорова-Успенського та ін. Розглянемо детальніше лише три — машини Поста, машини Тьюрінга і нормальні алгорифми Маркова.

3.2.1. Визначення і функціонування

Алгоритмічна система, запропонована в 1936 році. Е. Постом і названа **машинами Поста**, являє собою спеціальний клас алгоритмів. Для машин Поста вхідна й вихідна інформація задаються в ал-

фавіті $X = \{0, 1\}$, а алгоритм — у вигляді скінченного упорядкованого набору правил (команд), які називаються **наказами**. Для запису вхідної, вихідної та проміжної інформації служить гіпотетична нескінченна в обидві сторони **інформаційна стрічка**, поділена на окремі клітинки, у кожній з яких можна розмістити лише одну літеру: 0 або 1. Ті клітинки, в яких записані одиниці, називаються **відміченими**, а ті, в яких записані нулі, — **невідміченими**. Крім того, існує ще один «пустий» символ #, яким заповнюються всі клітинки інформаційної стрічки, що не зайняті вхідною інформацією.

Робота алгоритму проходить дискретними кроками, на кожному з яких виконується один з наказів, що складають алгоритм. Кожному кроку відповідає цілком визначена *активна клітинка* на інформаційній стрічці. Для першого наказу алгоритму фіксується як активна деяка *початкова* клітинка. Подальші зміни місцезнаходження активної клітинки на стрічці мають бути передбачені в самому алгоритмі. Накази, які складають алгоритм, належать до одного з таких шести типів.

Перший тип. Відмітити активну клітинку (записати в неї одиницю) і перейти до виконання i-го наказу (i може бути довільним числом із чисел, які використані для нумерації наказів алгоритму).

Другий тип. Стерти відмітку активної клітинки (записати в неї нуль) і перейти до виконання i-го наказу.

Третій тип. Змістити активну клітинку на один крок вправо і перейти до виконання i-го наказу.

Четвертий тип. Змістити активну клітинку на один крок вліво і перейти до виконання i-го наказу.

П'ятий тип. Якщо активна клітинка відмічена (якщо в ній записана одиниця), то перейти до виконання j-го наказу, а якщо не відмічена (якщо в ній записаний нуль), то перейти до виконання i-го наказу.

Шостий тип. Зупинка, закінчення роботи алгоритму.

Алгоритми, які складаються із скінченного числа наказів перелічених типів, називаються **алгоритмами Поста**. Отже, для того щоб задати машину Поста, необхідно задати таку четвірку ($X = \{0, 1\}$: алфавіт машини, початкову активну клітинку машини, # — спеціальний символ, P — алгоритм функціонування машини.

Робота машини Поста полягає у виконанні наказів алгоритму функціонування машини. Машина зупиняється тоді і тільки тоді, коли останнім виконаним наказом машини є наказ шостого типу і **результатом її роботи є слово** q, яке записане на інформаційній стрічці і яке називається **заключним**. Отже, машина Поста перероблює висхідні слова у заключні, тобто обчислює значення деякої слов-

никової функції f_R у алфавіті X, для якої висхідні слова є її аргументами, а заключне слово — значенням цієї функції. Якщо машина Поста зупиняється з деяким словом на інформаційній стрічці, то функція, що визначена цією машиною, вважається визначеною. А якщо машина Поста не зупиняється, то значення відповідної функції вважається невизначеним. За інтерпретації заключного слова як значення функції пусті символи # ігноруються.

Доведено, що алгоритм Поста обчислює деяку частково рекурсивну функцію, і, навпаки, довільна частково рекурсивна функція може бути обчислена відповідним алгоритмом Поста. Інакше кажучи, має місце таке твердження.

Теорема 60. *Клас усіх алгоритмів, еквівалентних алгоритмам Поста, збігається з класом усіх частково рекурсивних функцій [26, 31].*

Приклад алгоритму Поста

Покажемо, що найпростіша функція $s(n) = n + 1$ обчислюється на машині Поста. Для цього подамо число n у двійковій системі числення, тобто в алфавіті $X = \{0, 1\}$, запишемо його на інформаційну стрічку і встановимо активну клітинку, в якій знаходиться перший символ числа n (його старший двійковий розряд). Тоді алгоритм P функціонування машини Поста має такий вигляд:

1. Якщо в активній клітинці 0, то перейти до виконання 2, інакше перейти до виконання 3.
2. Змістити активну клітинку на один крок праворуч і перейти до виконання 1.
3. Якщо в активній клітинці 1, то перейти до виконання 2, інакше перейти до виконання 4.
4. Змістити активну клітинку на один крок ліворуч і перейти до виконання 5.
5. Якщо в активній клітинці 0, то перейти до виконання 6, інакше перейти до виконання 7.
6. Записати в активну клітинку 1 і виконати команду СТОП.
7. Записати в активну клітинку 0 і перейти до виконання 8.
8. Змістити активну клітинку на один крок ліворуч і перейти до виконання 9.
9. Якщо в активній клітинці 0, то перейти до виконання 6, інакше перейти до виконання 10.
10. Якщо в активній клітинці #, то перейти до виконання 6, інакше перейти до виконання 7.

Тестування цього алгоритму пропонується виконати як вправу. ♠

3.3. Застосування машин Поста до встановлення алгоритмічної повноти

У даному розділі описується застосування машин Поста до встановлення алгоритмічної повноти ЕОМ та процедурних мов програмування. Одне з найвагоміших досягнень сучасної техніки — це побудова універсальних електронних обчислювальних машин (ЕОМ), тобто таких автоматичних перетворювачів інформації, які дають можливість реалізовувати довільні алгоритми. Термін «універсальні» щодо цих машин розуміють як універсальність по відношенню до обчислювальних алгоритмів.

Оскільки довільний алгоритм може бути зведений, як уже зазначалося, до обчислення значення деякої частково рекурсивної функції, то універсальність по відношенню до обчислювальних алгоритмів є універсальністю взагалі. З'ясуємо питання про те, що повинна вміти робити система забезпечення роботи ЕОМ, щоб гарантувати цю універсальність.

Принцип керування, який забезпечує алгоритмічну універсальність ЕОМ, являє собою розвиток і узагальнення принципу, покладеного в основу алгоритмічної системи Поста. Як і в машині Поста, інформація в ЕОМ зберігається в пам'яті, яка розбита на чарунки (чарунки пам'яті). Щоправда, на відміну від машин Поста в кожній чарунці пам'яті може зберігатися не одне із чисел 0 чи 1, а ціле слово в алфавіті 0, 1, яке може складатися із значної (як правило, 30—40) кількості двійкових цифр 0 та 1. У зв'язку з цим розглядають вміст кожної чарунки пам'яті не як одну літеру алфавіту, а як слово (інформаційне слово) у двійковому алфавіті 0, 1. Двійкові цифри, які входять у запис слова, називаються двійковими розрядами, а саме слово — двійковим кодом, або просто двійковим числом.

Крім заміни двійкових цифр багаторозрядними двійковими кодами, існує ще одна істотна відмінність в організації пам'яті універсальної ЕОМ (УЕОМ) і пам'яті машини Поста. Як абстрактна алгоритмічна система, машина Поста є системою з необмеженим обсягом пам'яті, у той час як реальні технічні пристрої, такі як УЕОМ, мають обмежені обсяги пам'яті. Маючи на увазі те, що нескінченну пам'ять неможливо реалізувати в жодному технічному пристрої, прийнято називати машину універсальною, якщо організація її керування і набір операцій такі, що дають можливість реалізувати будь-який алгоритм за умови необмеженого обсягу пам'яті.

Універсальність сучасних ЕОМ на практиці забезпечується тим, що крім швидкодіючої (оперативної) пам'яті, ЕОМ забезпечується

відносно повільними (зовнішніми) пристроями пам'яті, які можуть обмінюватися інформацією з оперативним пристроєм пам'яті. Обсяг зовнішньої пам'яті можна вважати майже необмеженим, що й забезпечує (за наявності засобів обміну кодами з оперативною пам'яттю) практичну можливість реалізації будь-якого алгоритму.

Послідовність операцій, що їх виконує УЕОМ, задається за допомогою програми, яка закладається в пам'ять ЕОМ і являє собою упорядковану скінченну множину наказів (команд). Ці накази можна розглядати як узагальнення наказів, які використовуються у побудові алгоритмічної системи Поста. На відміну від системи Поста, в якій під час виконання кожного наступного наказу активна чарунка переміщується не більше ніж на один крок ліворуч чи праворуч, в УЕОМ передбачена можливість довільної зміни положення активної чарунки від наказу до наказу. Для цього в кожний наказ вводиться номер однієї чи кількох чарунок пам'яті, які стають активними під час виконання даного наказу. Ці номери чарунок пам'яті в ЕОМ називають адресами. Число адрес, які можна використовувати в наказах сучасних ЕОМ, як правило, лежить у межах від 1 до 4. Відповідно до цього розрізняють одно-, дво-, три- і чотириадресні накази.

Розглянемо спочатку ЕОМ з чотириадресною системою організації наказів, тобто ЕОМ, у яких максимальне число адрес у наказі дорівнює чотирьом. Різні типи наказів відповідають різним операціям, що виконуються ЕОМ. Накази в ЕОМ записуються у вигляді двійкових чисел (як правило), які можуть зберігатися як в оперативній, так і в зовнішній пам'яті ЕОМ.

Будемо вважати, що в кожній чарунці пам'яті може зберігатися або командне слово (наказ), або інформаційне слово. Командне слово поділяється на операційну частину та адресну. У першій частині стоїть код операції, яка виконується під час дії наказу, що зображається цим командним словом. У другій — розміщуються адреси чарунок, активних під час дії наказу.

Операції, які виконує ЕОМ, поділяються на кілька класів. До першого класу належать арифметичні операції — додавання, віднімання, множення та ділення. Чотириадресні накази для арифметичних операцій мають, як правило, таку структуру: в операційній частині наказу стоїть код операції — її умовний номер, який означає ту чи іншу дію (наприклад, 1 — додавання, 2 — віднімання, 3 — множення, 4 — ділення). Перші дві адреси в адресній частині наказу використовуються для задання адрес аргументів операції, яка вказана в наказі. Третя адреса в наказі служить для зберігання результату виконання даної арифметичної операції. Нарешті, четверта використо-

вується для задання адреси чарунки пам'яті, в якій зберігається наказ, що має виконуватися після діючого наказу.

Аналогічно будуються накази для виконання логічних операцій. Логічні операції здебільшого є бінарними і виконуються порозрядно, тобто окремо для кожної пари відповідних двійкових розрядів аргументів операції. До таких операцій належать порозрядна кон'юнкція і порозрядна диз'юнкція. Існують також і унарні логічні операції. До них належать ліві і праві зсуви, які перетворюють даний двійковий код $xy\ldots z$ на код $xy\ldots 0$ і $0xy\ldots z$ відповідно.

Особливу роль відіграють так звані операції передачі керування, які служать для зміни порядку виконання наказів програми залежно від результатів, що одержані в процесі її виконання. Типовою операцією передачі керування (яку часто називають операцією умовного переходу) є операція умовного переходу за точним збігом слів. Перші дві адреси в наказі, що реалізує дану операцію, використовуються для чарунок пам'яті, з яких вибираються слова, що порівнюються між собою. Якщо слова збігаються (рівні між собою), наступний наказ береться з чарунки, яка знаходиться за третьою адресою, якщо ж слова не збігаються — за четвертою адресою в наказі умовного переходу. Можливі й інші умовні переходи, наприклад умовний перехід залежно від знака різниці двох слів або залежно від знака одного якогонебудь слова (в останньому випадку очевидно, що в наказі умовного переходу досить розмістити три, а не чотири адреси).

Пам'ять ЕОМ влаштована таким чином, що при читанні слова з якої-небудь чарунки пам'яті для виконання тієї чи іншої операції виникає копія цього слова, яка надходить у відповідний пристрій для виконання операції, а саме слово залишається в чарунці, з якої воно було прочитане (скопійоване). Якщо виконується запис нового слова в ту чи іншу чарунку пам'яті, то інформація, яка була в цій чарунці до запису, автоматично знищується (стирається).

Беручи до уваги всі ці властивості пам'яті, неважко помітити, що для виконання довільного алгоритму Поста достатньо користуватися лише операцією (алгебраїчного) додавання, однією з операцій умовного переходу за точним збігом слів та операцією зупинки машини.

Дійсно, умовимося оперувати лише двома будь-якими інформаційними словами — p і q, перше з яких ототожнимо з нулем, а друге — з одиницею двійкового алфавіту будь-якого заданого алгоритму Поста A. Пам'ять УЕОМ M поділимо на три частини. Перша складається всього із п'яти чарунок: a_{-1}, a_0, a_1, b_0 і b_1, в яких розміщені числа –1, 0, 1, p, q; у чарунках другої частини розмістимо програму алгоритму A; нарешті, третя частина імітує інформаційну стрічку алгоритму A.

Перейдемо до безпосереднього моделювання наказів алгоритму Поста A наказами машини M.

Як відомо, алгоритми Поста використовують шість різних типів наказів. Наказ шостого типу безпосередньо моделюється відповідним наказом машини M. Накази перших двох типів (запис у чарунку числа 0 або 1) моделюються наказами машини M, які виконують пересилання інформації із чарунок b_0 чи b_1 в активну чарунку, вказану, наприклад, у третій адресі машинного наказу. Зрозуміло, що таке пересилання можна виконати наказом додавання, у перших двох адресах якого стоїть пара адрес чарунок a_0 і b_0 або a_0 і b_1, а в третій — адреса активної чарунки (четверта адреса, як і в машині Поста, використовується для розміщення адреси наказу, який повинен виконуватися наступним після діючого).

Неважко помітити, що постівський наказ п'ятого типу моделюється машинним наказом умовного переходу за точним збігом слів. Для цього достатньо порівняти слово в чарунці b_1 зі словом в активній чарунці і перейти залежно від результатів цього порівняння до одного з двох наказів.

Нарешті, довільний постівський наказ третього чи четвертого типів моделюється за допомогою групи машинних наказів, число яких дорівнює числу наказів першого, другого і п'ятого типів у заданій програмі алгоритму Поста A.

Дійсно, нехай r_i — довільний наказ першого, другого чи п'ятого типу алгоритму A. Згідно із зазначеним вище він моделюється одним машинним наказом, який позначимо r. Нехай постівський наказ q_i зміщує активну чарунку на одну одиницю праворуч. Для машинної програми таке зміщення може бути виконане лише за рахунок зміни на +1 усіх адрес активних чарунок. Такі чарунки трапляються лише в наказах машини, яка моделює постівські накази першого, другого та п'ятого типів. За рахунок підходящої нумерації адрес, не обмежуючи загальності, можна допустити, що адреси активних постівських чарунок записуються в будь-якій заданій, наприклад останній, адресі в наказі. Якщо вважати адреси цілими двійковими числами, то доходимо висновку, що для зміни адреси активної чарунки в наказі r на +1 достатньо додати до її адреси в команді r число 1, розміщене у відповідній адресі чарунки a_1. Аналогічно виконується зміщення ліворуч.

Із зазначеного випливає, що алгоритмічна універсальність будь-якого цифрового автомата, який працює під керуванням програми, буде забезпечена, якщо за допомогою операцій, які виконуються цим автоматом, можна виконати такі чотири операції:

1) операцію пересилання вмісту будь-якої чарунки пам'яті в будь-яку іншу чарунку пам'яті;

2) операцію додавання констант до командного слова (наказу), розміщеного в будь-якій чарунці пам'яті, з метою зміни величини заданої адреси в наказі на 1 або –1;

3) операцію умовного переходу за точним збігом слів;

4) операцію (безумовної) зупинки машини.

Очевидно також, що крім перелічених операцій у наборі кожної ЕОМ мають бути також операції введення (виведення) інформації в машину (із машини).

Звідси випливають також умови функціональної повноти, які має задовольняти універсальна мова програмування.

Розглянемо тепер питання про шляхи зменшення кількості адрес у наказах. Перш за все неважко зрозуміти, що четверта адреса в наказі p може бути вилучена за рахунок такого слідування наказів у пам'яті ЕОМ: адреса наказу p', який повинен виконуватися наступним, має бути на одиницю більшою адреси самого наказу p. Інакше кажучи, використання четвертої адреси стає непотрібним за умови, що порядок слідування наказів у чарунках пам'яті відповідає порядку їх виконання машиною.

Зміна послідовного порядку виконання наказів можлива у разі, коли виконується наказ умовного переходу. Як зазначалося вище, у чотириадресній системі умовних наказів одна з адрес використовується для задання наступного наказу у разі невиконання заданої умови, а інша — для задання наступного наказу у разі виконання цієї умови. У разі заміни чотириадресної системи наказів триадресною приймають, що в першому випадку (за невиконання умови) після наказу умовного переходу виконується наказ, записаний у наступній по порядку чарунці пам'яті, і лише в другому випадку, тобто при виконанні умови, одна з адрес в умовному наказі використовується для задання адреси наказу, який повинен виконуватися наступним.

Із сказаного випливає, що четверта адреса в наказі може стати зайвою не тільки в звичайних наказах, які не змінюють подальшого порядку виконання наказів, але і в наказах умовних переходів. Отже, ми приходимо до триадресної системи наказів.

Подальше зменшення числа адрес у наказах може бути досягнуто за рахунок фіксації деякої допоміжної чарунки пам'яті, конструктивно відокремленої від інших чарунок ЕОМ. Після фіксації такої чарунки відкривається простий шлях, який дає змогу зменшити число адрес у наказі до мінімуму, тобто до однієї адреси. Пояснимо це на прикладі операції додавання, що вимагає трьох адрес: першого аргументу a, другого аргументу b і адреси c для зберігання суми. За допомогою фіксованої чарунки d цю триадресну операцію можна виконати шляхом послідовного виконання трьох одноадресних операцій — пересилання

числа a у фіксовану чарунку d, додавання числа, яке міститься в чарунці b, до числа в чарунці d і, нарешті, пересилання числа із чарунки d в чарунку з адресою c. Оскільки при цьому можуть змінюватися лише адреси a, b, c, а адреса d раз і назавжди фіксована, то всі три вказані операції реалізуються за допомогою одноадресних наказів.

Описаний спосіб приводить нас до одноадресної системи наказів. Маючи одноадресну систему наказів, неважко побудувати також двоадресну систему наказів. Друга адреса при цьому може використовуватися як для задання адреси наступного наказу, так і для задання числових кодів (інформаційних слів), над якими виконуються операції.

3.4. Машини Тьюрінга

3.4.1. Визначення і функціонування

До алгоритмічної системи Поста дуже близька система Тьюрінга, яку називають **машинами Тьюрінга.** У цій системі також виконується запис інформації на розбитій на клітинки нескінченній в обидва боки інформаційній стрічці. Але, на відміну від системи Поста, тут для запису інформації використовується довільний скінченний алфавіт $X = \{x_1, x_2, \ldots, x_n\}$. Кожна клітинка інформаційної стрічки служить для запису лише одного символу. Цей символ може оглядатися спеціальним чутливим елементом так званої **головки Г машини Тьюрінга**, яка має змогу пересуватися вздовж інформаційної стрічки в обидва боки. Головка машини Тьюрінга може знаходитись у скінченному числі різних станів $q_1, q_2, \ldots, q_m$, друкувати в клітинці, яку оглядає головка, довільний символ $x_1, x_2, \ldots, x_n$ і залишатися на місці або переміщуватися вправо чи вліво вздовж інформаційної стрічки на одну клітинку (див. рис. 3.4.1).

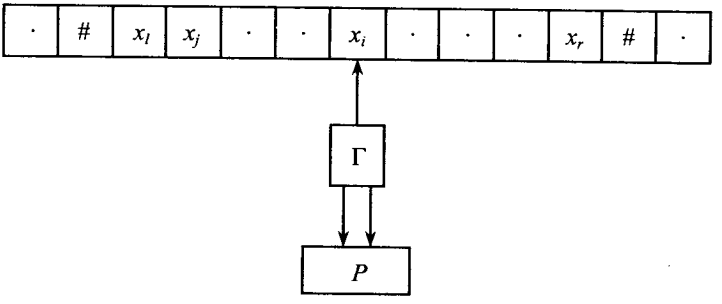


Рис. 3.4.1. Схема машини Тьюрінга

Запис алгоритму, який реалізується машиною Тьюрінга, виконується за допомогою програми P роботи цієї машини. **Програма роботи машини Тьюрінга** — це скінченна множина п'ятірок символів вигляду $x_i q_j x_k q_r s_p$. Кожна така п'ятірка називається **командою** і виконання команди $x_i q_j x_k q_r s_p$ означає, що коли головка машини Тьюрінга знаходиться в стані q_j і оглядає записаний на стрічці символ x_i, то вона записує на місце цього символа новий символ x_k (який може збігатися з раніше записаним — x_i), переходить у новий стан q_r (який може збігатися з попереднім станом), а головка переміщується вздовж стрічки на величину $s_p = 0, \pm 1$.

Отже, визначити машину Тьюрінга означає задати таку п'ятірку об'єктів $(X, Q, q_0, \#, P)$, де

X — алфавіт машини;

Q — скінченна непуста множина станів $(X \cap Q = \varnothing)$;

q_0 — початковий стан машини $(q_0 \in Q)$;

$\#$ — спеціальний «пустий» символ, такий що $\# \notin X \cup Q$;

P — програма роботи машини.

Робота машини Тьюрінга полягає в повторенні такого циклу дій, які є спільними для довільної машини Тьюрінга:

а) **читання символу з клітинки**, яку оглядає головка;

б) **пошук застосовної команди**, тобто пошук команди $x_i q_j x_k q_r s_p$, у якої q_j — стан головки в даний момент часу, а x_i — символ у клітинці, яку оглядає головка, знаходячись у стані q_j;

в) **виконання знайденої команди**.

При цьому вважається, що в програмі P не існує ніяких двох команд з однаковими першими символами.

Машина Тьюрінга зупиняється тоді і тільки тоді, коли ні одна з команд її програми не застосовна. **Результатом роботи машини** після її зупинки **є слово, записане на інформаційній стрічці**.

Після цих пояснень машина Тьюрінга стає алгоритмічною системою. При цьому машина Тьюрінга обробляє заздалегідь записане на стрічці слово або нескінченно довго, або зупиняється після скінченного числа кроків. У першому випадку вважається, що алгоритм, який виконується машиною, не застосовний до вхідного слова p, а в другому випадку слово, яке залишається на стрічці після зупинки машини, приймається як вихідне слово, до якого машина перетворює задане вхідне слово p. Це слово називається **заключним** і є результатом роботи машини Тьюрінга.

Машина Тьюрінга, перероблюючи початкові слова в заключні, визначає деяку словникову функцію, для якої початкові слова є аргументами, а заключне слово — значенням цієї функції. Зрозуміло, що робота машини може закінчуватися через деякий час, а може й

ніколи не закінчуватись. Якщо машина Тьюрінга зупиняє свою роботу з деяким словом на стрічці, то функція, яка визначається цією машиною, вважається визначеною, а якщо машина не зупиняється, то значення відповідної функції вважається невизначеним. При інтерпретації заключного слова як значення функції пусті символи # ігноруються. Таким чином, машина Тьюрінга T задає деяку часткову функцію f_T і спосіб її обчислення.

Функція f називається частково обчислюваною за Тьюрінгом, якщо існує така машина Тьюрінга T, що $f = f_T$. У цьому випадку також говорять, що для функції f існує (частковий) алгоритм обчислення її значень, який визначається машиною T.

Для машин Тьюрінга, як і для машин Поста, має місце така теорема.

Теорема 61. *Довільна частково рекурсивна функція обчислювана за Тьюрінгом і, навпаки, довільна машина Тьюрінга обчислює деяку частково рекурсивну функцію [26, 28].*

Машина Тьюрінга, як і машина Поста, являє собою формальну модель алгоритму. Дійсно, для них характерні такі властивості, як дискретність, детермінованість, елементарність кроків, направленість і масовість (дивись інтуїтивне визначення алгоритму). Зазначимо також, що для цих машин характерні:

а) *конструктивність* — машина являє собою скінченний об'єкт, побудований за цілком визначеними правилами;

б) *скінченність* — процес визначення значень функції f_T (f_R), для яких вона визначена, складається із скінченного числа кроків;

в) *однозначність* — результат роботи машини однозначно визначається початковим словом.

Приклади. а) Нехай $(X = \{0, 1\}, Q = \{q_0, q_1, q_2\}, q_0, \#, P)$, де P має такий вигляд:

$$1q_01q_0 + 1$$
$$0q_00q_0 + 1$$
$$\#q_0\#q_1 - 1$$
$$1q_10q_1 - 1$$
$$0q_11q_2 - 1$$
$$\#q_11q_2 - 1,$$

де в початковий момент часу головка оглядає перший символ слова $p \in F(X)$ у стані q_0.

Неважко переконатися в тому, що ця машина реалізовує алгоритм обчислення функції $s(n) = n + 1$, аргументи і результат якої

представлені в двійковій системі числення. Протестуємо цей алгоритм. Нехай $p = 100111$, тоді маємо таку послідовність команд:

$$1q_0 1q_0 + 1 \to 0q_0 0q_0 + 1 \to 0q_0 0q_0 + 1 \to 1q_0 1q_0 + 1 \to 1q_0 1q_0 + 1 \to 1q_0 1q_0 + 1 \to$$
$$\to \#q_0 \#q_1 - 1 \to 1q_1 0q_1 - 1 \to 1q_1 0q_1 - 1 \to 1q_1 0q_1 - 1 \to 0q_1 1q_2 - 1.$$

б) Нехай $(X = \{0, 1\}, Q = \{q_0, q_1\}, q_0, \#, P)$, де P має такий вигляд:

$$1q_0 1q_1 - 1$$
$$0q_0 0q_1 - 1$$
$$\#q_1 1q_1 - 1.$$

Дія цієї машини, як неважко переконатися, полягає в тому, що почавши роботу на довільному непустому слові p, до цього слова приписується зліва символ 1. При цьому машина ніколи не зупиняється. Отже, результат роботи машини не визначений, оскільки алгоритм не застосовний до довільного непустого слова $p \in F(X)$.

в) Нехай $(X = \{0, 1\}, Q = \{q_0, q_1\}, q_0, \#, P)$, де P має такий вигляд:

$$\#q_0 \#q_0 + 1$$
$$0q_0 0q_0 + 1$$
$$1q_0 1q_1 0.$$

Дія цієї машини зводиться до пошуку в слові $p \in F(X)$ символу 1 і зупинки, після того як цей символ знайдений.

Якщо слово p є записом у двійковій системі числення деякого натурального числа n, то алгоритм цієї машини застосовний до довільного відмінного від нуля числа. До числа нуль цей алгоритм не застосовний.

г) Нехай $X = \{(,), 0, 1, *\}$ і $p \in F(X)$, де $F(X)$ складають ті і тільки ті слова в алфавіті X, які побудовані з перших двох символів X. Правильними слова називаються слова вигляду

г1) e — пусте слово,

г2) якщо α — правильне слово, то слова (α), $()\alpha$, $\alpha()$ — теж правильні слова.

Нехай $M \subseteq F(X)$ означає множину правильних слів. Побудуємо машину T, на стрічку якої записуються слова із $F(X)$, і T зупиняється з результатом 1 на стрічці, якщо слово $p \in M$, і з результатом 0 у протилежному випадку. Іншими словами, T повинна обчислювати значення предиката π: $F(X) \to \{0, 1\}$ такого, що

$$\pi(p) = \begin{cases} 1, & \text{якщо } p \in M, \\ 0, & \text{якщо } p \notin M. \end{cases}$$

Шукана машина Тьюрінга має такий вигляд:

$$T = (\{(,), 0, 1, *\}, \{q_0, q_1, q_2, q_3, q_4\}, q_0, \#, P),$$

де програма P має такі команди (щоб не виникло плутанини, символу s відповідає символ 0, символу r — +1, символу l — –1):

1. $(q_0(q_0r,$ 2. $)q_0 * q_1l,$
3. $*q_0 * q_0r:$ 4. $\#q_0\#q_2l,$
5. $(q_1 * q_0r,$ 6. $*q_1 * q_1l,$
7. $\#q_10q_4r,$ 8. $(q_2\#q_3l,$
9. $*q_2\#q_3l,$ 10. $\#q_21q_2s,$
11. $(q_3\#q_3l,$ 12. $*q_3\#q_3l,$
13. $\#q_30q_3s,$ 14. $(q_4\#q_4r,$
15. $)q_4\#q_4r,$ 16. $*q_4\#q_4r.$

Розглянемо роботу машини T на словах $p_1 = (())$, $p_2 = (()$. Маємо

$$T(p_1) = (q_0()) = ((q_0)) = (()q_0) = ((q_1*) = (* * q_0) = (**)q_0 =$$
$$= (* * q_1 * = (*q_1 * * = (q_1 * * * = *q_0 * * * = * * q_0 * * = * * *q_0 * =$$
$$= * * * * q_0\# = * * * * \#q_0 = * * *q_2 = * * q_2 = *q_2 = \#q_2 = q_21.$$

Таким чином, ні одна з команд T не застосовна до слова 1, яке записане на стрічці, і отже, слово 1 — результат роботи машини T.

$$T(p_2) = (q_0() = ((q_0) = (()q_0 = ((q_1* = (*q_0* = (* * q_0\# =$$
$$= (* * \#q_0 = (*q_2 = (q_2 = \#q_3 = q_30.$$

Отже, результатом роботи є слово 0. ♠

Функціонування машини Тьюрінга можна описати за допомогою протоколу роботи над заданим початковим словом. Нехай $\#y_1 \ldots y_k y_{k+1} \ldots y_m\#$ — слово, що виникає на стрічці в процесі роботи машини після виконання деякої команди машини, у результаті якої машина знаходиться в стані q, а головка стоїть навпроти клітинки стрічки, де записаний k-й символ слова. Слово $\#y_1 \ldots y_k q y_{k+1} \ldots y_m\#$ називається **конфігурацією машини** T. Послідовність конфігурацій, записаних у тому порядку, в якому вони появляються в процесі роботи машини, називається **протоколом роботи машини** T. У наведеному вище прикладі процес роботи машини Тьюрінга описувався за допомогою протоколів.

3.4.2. Словникове представлення машини Тьюрінга

Машина Тьюрінга однозначно задається своєю програмою. Якщо упорядкувати деяким способом її команди і застосувати спосіб кодування послідовності команд словом в алфавіті машини Тьюрінга, який описано нижче, то можна одержати її представлення у власному алфавіті.

Нехай X — алфавіт машини Тьюрінга T, а Q — множина її станів. Упорядкуємо деяким чином множину Q, і нехай $K(q)$ порядковий номер стану q. Введемо додатковий символ $*$, який не входить в X, і співставимо кожній команді $aqa'q'd$ слово в алфавіті $W = X \cup \cup \{\#, l, r, s, *\}$:

$$a * {}^{K(q)} a' * {}^{K(q')} d.$$

Упорядкованій послідовності команд відповідає послідовність слів у алфавіті W. Результатом конкатенації є слово p_T, яке однозначно описує машину T. Наступний етап кодування — це перехід від представлення машини T в алфавіті W до представлення її в алфавіті X. Якщо алфавіт X має n символів, то упорядкувавши його якимнебудь способом, упорядкуємо алфавіт W так, щоб додаткові символи, які не входять в X, отримали такі номери: $K'(\#) = n + 1$, $K'(*) = n + 2$, $K'(l) = n + 3$, $K'(r) = n + 4$, $K'(s) = n + 5$.

Закодувавши слова із $F(X)$ і $F(W)$ числами і використовуючи лексикографічний порядок, знайдемо номер слова p_T із $F(W)$ і вибравши слово із $F(X)$ з тим же номером, одержимо словарне представлення α_T машини Тьюрінга T в її алфавіті. За словом α_T можна однозначно (з точністю до номерів станів) відновити програму машини T. Слід зазначити, що одній і тій самій машині Тьюрінга можуть відповідати різні словникові представлення в її алфавіті залежно від способу упорядкування множин Q, X і P, але за довільним із таких представлень програма машини відновлюється однозначно.

3.4.3. Алгоритмічно розв'язувані і нерозв'язувані проблеми

Нехай X — алфавіт, $M \subset F(X)$ — деяка множина слів у цьому алфавіті.

Характеристичною функцією множини M називається повністю визначений на $F(X)$ предикат

$$\xi : \mathrm{M} \to \{0, 1\},$$

який задається таким чином:

$$\xi(p) = \begin{cases} 1, & \text{якщо } p \in M, \\ 0, & \text{якщо } p \notin M. \end{cases}$$

Частковою характеристичною функцією множини M називається функція ξ: $M \to \{1\}$, яка визначається таким чином:

$$\xi(p) = \begin{cases} 1, & \text{якщо } p \in M, \\ \text{не визначено}, & \text{якщо } p \notin M. \end{cases}$$

Множина M називається **розв'язуваною**, якщо її характеристична функція обчислювана, тобто частково рекурсивна, і **перелічимою**, якщо її часткова характеристична функція обчислювана. Розв'язуваність множини M означає, що існує така машина Тьюрінга, яка завжди зупиняється і яка дає можливість за скінченне число кроків встановити, належить чи ні дане слово p із $F(X)$ множині M. Для перелічимої множини існує така машина Тьюрінга, яка зупиняється в тому і тільки в тому випадку, коли дане слово p належить множині M. У випадку перелічимої множини машина Тьюрінга не дає можливості точно встановити, належить чи ні задане слово множині M, оскільки відсутність відповіді до деякого моменту часу не несе ніякої інформації про те, з'явиться чи ні вона пізніше. Але коли машина зупинилася, то ми точно знаємо, що дане слово належить множині M.

Зв'язок між розв'язуваними і перелічимими множинами виявляє

Теорема 62 **(Теорема Поста).** *Множина $M \subseteq F(X)$ є розв'язуваною тоді і тільки тоді, коли M і її доповнення $M' = F(X) \setminus M$ перелічимі [26, 18].*

Доведення. Із розв'язуваності M випливає її перелічимість. Дійсно, машину Тьюрінга T_M, яка обчислює характеристичну функцію множини M, легко перетворити в машину Тьюрінга T, додавши до програми машини T команди, які зациклюють її в тому випадку, коли вона зупиняється з результатом 0. Нова машина задає часткову характеристичну функцію множини M.

Нехай T_M і T'_M — машини Тьюрінга, які визначають часткові характеристичні функції множин M і M'. Алгоритм, який розпізнає, належить чи ні слово p множині M, зводиться до такої процедури. Виконується по одній команді кожної із машин Тьюрінга з одним і тим самим словом p на стрічці. Якщо після виконання однієї команди зупиняється машина T_M, то результат роботи алгоритму — символ 1. Якщо зупиняється машина T'_M, то результат — символ 0. Якщо жодна з машин не зупиняється, то виконується наступна пара команд, і т.д. Оскільки p — це елемент або із M, або із M', то через скінченне число кроків або T_M, або T'_M зупиняться. Таким чином, алгоритм обчислює значення характеристичної функції множини M, і можна побудувати машину Тьюрінга T_M, яка реалізовує цей алгоритм. Отже, множина M розв'язувана. ■

Визначення розв'язуваної і перелічимої множини можна перенести і на числові множини, і множини, елементами яких є об'єкти

більш складної структури. Прикладами розв'язуваних множин можуть служити такі множини:

— пуста множина;

— множина всіх слів у деякому алфавіті X;

— множина всіх словникових представлень машин Тьюрінга над алфавітом X. Перейдемо тепер до знайомства з множинами, які не є розв'язуваними і перелічимими.

Проблема зупинки. Як відомо, машина Тьюрінга, працюючи над деяким словом p із $F(X)$, може як завершувати свою роботу, так і не завершувати її. Корисно було б мати алгоритм, який для довільного слова p із $F(X)$ з'ясовував, зупиниться чи ні машина Тьюрінга під час роботи зі словом p. Проблему зупинки можна сформулювати також у термінах множин. Нехай M_s — множина всіх пар слів у алфавіті X, де в кожній парі перше слово — це словникове представлення деякої машини Тьюрінга, а друге — таке слово, для якого машина, почавши роботу над ним, зупиняється. Проблема зупинки тепер має такий вигляд: чи є множина M розв'язуваною?

***Теорема 63* (Теорема Тьюрінга).** *Проблема зупинки машини Тьюрінга алгоритмічно нерозв'язувана.*

Доведення. Необхідно встановити, чи є обчислюваною характеристична функція $g_{M_s}: F(X)^2 \to \{0, 1\}$. Припустимо, що це так і T_g — машина Тьюрінга, яка обчислює цю функцію. З обчислюваності функції g_{M_s} і розв'язуваності множини M_t словникових представлень машин Тьюрінга в алфавіті X випливає обчислюваність часткової унарної функції G: $F(X) \to \{0, 1\}$, яка задається таким чином:

$$G(p) = \begin{cases} g_{M_s}, & \text{для всіх } p \in M_t, \\ \text{не визначено}, & \text{для всіх } p \notin M_t. \end{cases}$$

Отже, функція G зв'язує машини Тьюрінга зі своїми власними словниковими представленнями.

Введемо ще одну унарну функцію K: $F(X) \to \{0\}$, де

$$K(p) = \begin{cases} G(p), & \text{якщо } G(p) = 0, \\ \text{не визначено}, & \text{у протилежному випадку.} \end{cases}$$

Зрозуміло, що функція K обчислювана, якщо обчислювана функція G. Дійсно, машина Тьюрінга T_K повністю збігається з машиною Тьюрінга T_G, окрім випадку, коли машина T_G зупиняється: машина T_K продовжує роботу, щоб з'ясувати, який результат — 1 чи 0, і в другому випадку зупиняється, а в першому — зациклюється.

Нехай β_K — словникове представлення машини T_K в алфавіті X. Спробуємо з'ясувати, визначене чи ні значення $K(\beta_K)$, тобто спробуємо розв'язати проблему зупинки для пари — машини T_K і її словникового представлення. Припустимо, що значення $K(\beta_K)$ не визначено, тобто, що машина T_K не зупиняється, якщо вона почала роботу над словом β_K. Тоді $g_M(\beta_K, \beta_K) = 0$, $G(\beta_K) = 0$ і $K(\beta_K) = 0$, тобто значення $K(\beta_K)$ визначено. А це суперечить припущенню. Припустимо тепер, що значення $K(\beta_K)$ визначено, тобто машина T_K зупиняється, почавши роботу над словом β_K. Тоді $g_M(\beta_K, \beta_K) = 1$, $G(\beta_K) = 1$ і $K(\beta_K)$ не визначено, що також суперечить припущенню. Обидва припущення про функцію K приводять до суперечності, що спростовує гіпотезу про обчислюваність функції g_{M_s} і розв'язуваність множини M_s.■

Доведена теорема встановлює існування функцій, які не є обчислюваними (характеристична функція g_{M_s}). Із тези Черча випливає, що для такої функції не існує алгоритму обчислення її значень. Але ця теорема не виключає можливості, що проблема зупинки може виявитися розв'язуваною для деякого більш вузького класу машин Тьюрінга.

Проблема пустої стрічки. Розглянемо окремий випадок проблеми зупинки. З'ясуємо, чи є розв'язуваною проблема зупинки машини Тьюрінга, яка застосовується до пустої стрічки, тобто до стрічки, яка включає лише пустий символ #. Іншими словами, необхідно з'ясувати, є чи ні розв'язуваною множина M_s словникових представлень усіх машин Тьюрінга, які зупиняються, почавши роботу при пустій стрічці.

Теорема 64. *Проблема пустої стрічки алгоритмічно нерозв'язувана [18].*

Доведення. Кожній парі (T, p), де T — деяка машина Тьюрінга, а p — слово в її алфавіті, надамо у відповідність машину T_p, програма якої збігається з програмою машини T, крім того, що почавши роботу, машина T_p стирає початкове слово на стрічці і записує замість нього слово p. Конструкція машини T_p очевидна: до програми машини T необхідно додати відповідним чином команди машини із вправи 15 (див. вправи у кінці параграфа).

Машина T_p, почавши роботу з пустою стрічкою, веде себе після запису слова p так, як і машина T, що застосовується до слова p. Зокрема, машина T_p зупиниться в тому і тільки в тому випадку, коли зупиниться машина T. Припустимо, що проблема зупинки машини Тьюрінга з пустою стрічкою розв'язувана. Але тоді звідси випливала б розв'язуваність проблеми зупинки в загальному випадку. Дійс-

но, для того щоб дізнатися, зупиняється чи ні деяка машина T, яка застосовується до слова p, необхідно сконструювати машину T_p. З'ясувавши, зупиняється чи ні T_p при пустій стрічці, ми тим самим дізналися б, зупиняється чи ні машина T. А це суперечить теоремі Тьюрінга і спростовує припущення про розв'язуваність проблеми пустої стрічки. ■

При доведенні цієї теореми використовувалась така схема: виконавши деякі допоміжні побудови, ми припускаємо розв'язуваність проблеми, яка досліджується, і цим одержуємо можливість розв'язувати іншу проблему, про яку відомо, що вона не є розв'язуваною. Одержана суперечність доводить нерозв'язуваність проблеми, що досліджується. Такий спосіб доведення називається **методом зведення**: відома нерозв'язувана проблема зводиться до проблеми, яка нас цікавить, а остання не є розв'язуваною. Цей метод також застосовується і у випадку, коли доводиться, що проблема, яка нас цікавить, не є частково розв'язуваною. При використанні цього методу будується послідовність проблем, які зводяться одна до другої, і часто в цій послідовності «базовою» нерозв'язуваною проблемою є проблема зупинки машин Тьюрінга. Слід зазначити, що метод зведення не є універсальним: існують нерозв'язувані проблеми, які взаємно не зводяться одна до другої.

Іншою «базовою» проблемою, що часто використовується в доведеннях методом зведення, є **проблема тотожності слів** або **проблемою відповідностей Поста** [18].

Нехай $V = \{p_1, p_2, \ldots, p_n\}$ і $U = \{q_1, q_2, \ldots, q_n\}$ — два кортежі слів у алфавіті X однакової довжини. Пару V, U називають **системою Поста.** Непусту скінченну послідовність індексів $i_1, i_2, \ldots, i_k$, $1 \le k \le n$ називають розв'язком системи Поста, якщо $p_{i_1} p_{i_2} \ldots p_{i_k} = q_{i_1} q_{i_2} \ldots q_{i_k}$, де зліва і зправа стоять слова, отримані в результаті конкатенації відповідних слів, що вибрані із V і U. Чи існує алгоритм, який відповідає на питання: має чи ні розв'язок система Поста? Пост показав нерозв'язуваність цієї проблеми для алфавіту, який містить більше одного символу. У той же час відомо, що проблема Поста є частково розв'язуваною.

Проблема зациклювання. Наведемо тепер приклад, який стверджує існування неперелічимих множин і проблем, які не є частково розв'язуваними. Проблема зациклювання формулюється так: чи існує алгоритм (хоча б частковий), який визначає наперед для довільної машини Тьюрінга і довільного слова p, буде чи ні дана машина працювати над словом p нескінченно довго, тобто необхідно встановити, буде чи ні розв'язуваною або перелічимою множина $M_c \subset F(X)^2$ усіх пар слів таких, що перше слово — словникове представлення машини Тьюрінга, а друге — слово, на якому ця машина зациклюється.

Теорема 65. *Проблема зациклювання машин Тьюрінга не є частково розв'язуваною проблемою.*

Доведення цієї теореми опускається, оскільки виходить за рамки даного курсу. За необхідності його можна знайти, наприклад, у [18].

3.5. Контрольні питання, задачі і вправи

Контрольні питання

1. Які функції називаються найпростішими?
2. Дайте визначення операцій суперпозиції, примітивної рекурсії і мінімізації.
3. Яка функція називається примітивно рекурсивною, частково рекурсивною?
4. Сформулюйте тезу Черча, тезу Тьюрінга.
5. Які команди виконуються в алгоритмічній системі Поста?
6. Чим відрізняється алгоритмічна система Поста від системи Тьюрінга?
7. Яку роботу виконує головка машини Тьюрінга в процесі роботи цієї машини?
8. Яка функція називається характеристичною функцією множини $M \subseteq F(X)$?
9. Дайте визначення часткової характеристичної функції.
10. Дайте визначення множини, яка розв'язувана і перелічима.

Задачі і вправи

1. Довести, що
   a) $x \div y = s(x) \div s(y)$,
   b) $x + (y \div x) = y + (x \div y)$,
   c) $x \div (y + z) = (x \div y) \div z$,
   d) $(x \div y) \div z = (x \div z) \div y$.
2. Довести, що якщо функція $f(x_1, x_2, \ldots, x_n)$ примітивно рекурсивна, то наведені нижче функції теж примітивно рекурсивні:
   a) $g(x_1, x_2, x_3, \ldots, x_n) = f(x_2, x_1, x_3, \ldots, x_n)$; (перестановка аргументів);
   b) $g(x_1, x_2, \ldots, x_n) = f(x_2, x_3, \ldots, x_n, x_1)$; (циклічна перестановка аргументів);
   c) $g(x_1, x_2, \ldots, x_n, x_{n+1}) = f(x_1, x_2, \ldots, x_n)$ (введення фіктивного аргументу);
   d) $g(x_1, x_2, \ldots, x_{n-1}) = f(x_1, x_1, x_2, \ldots, x_{n-1})$ (ототожнення аргументів).

3. Довести, що функція $f(x) = x!$ — примітивно рекурсивна.

4. Які функції можна побудувати з функцій f і h за допомогою операції примітивної рекурсії:

a) $f(x) = x$, $h(x, y, z) = z^x$; b) $f(x) = x$, $h(x, y, z) = x^z$?

5. Довести, що наведені нижче функції примітивно рекурсивні:

$$\text{a) } sg(x) = \begin{cases} 0, & \text{якщо } x = 0, \\ 1, & \text{якщо } x > 0; \end{cases}$$

$$\text{b) } \overline{sg}(x) = \begin{cases} 1, & \text{якщо } x = 0, \\ 0, & \text{якщо } x > 0; \end{cases}$$

c) $\max(x, y)$; *d*) $\min(x, y)$.

6. Довести, що нижченаведені функції примітивно рекурсивні:

a) $\tau(x)$ — число дільників числа x, де $\tau(0) = 0$;

b) $\sigma(x)$ — сума дільників числа x, де $\sigma(0) = 0$;

c) $lh(x)$ — число простих дільників числа x, де $lh(0) = 0$;

d) $\pi(x)$ — число простих чисел, не більших від x;

e) НСК(x, y) — найменше спільне кратне чисел x і y, де НСК(x, 0) = НСК(0, y) = 0;

f) НСД(x, y) — найбільший спільний дільник чисел x і y, де НСД(0, 0) = 0.

7. Довести, що наведені нижче функції частково рекурсивні:

a) ніде не визначена функція ε;

$$\text{b) } f(x,y) = \begin{cases} x - y, & \text{якщо } x \geq y, \\ \text{невизначено} & \text{в решті випадків;} \end{cases}$$

$$\text{c) } f(x,y) = \begin{cases} x / y, & \text{якщо } y \text{ ділить } x, \\ \text{невизначено} & \text{в решті випадків;} \end{cases}$$

$$\text{d) } f(x,y) = \begin{cases} z, & \text{якщо } z^y = x, \\ \text{невизначено} & \text{в решті випадків;} \end{cases}$$

e) функція визначена лише в скінченному числі точок.

8. Довести, що коли z, g, h частково рекурсивні функції, то частково рекурсивними будуть і такі функції [23]:

a) $\mu_y[z(x_1, x_2, \ldots, x_n, y) = g(x_1, x_2, \ldots, x_n, y)]$;

b) $\mu_y[z(x_1, x_2, \ldots, x_n, y) \neq g(x_1, x_2, \ldots, x_n, y)]$;

c) $\mu_y[z(x_1, x_2, \ldots, x_n, y) \leq g(x_1, x_2, \ldots, x_n, y)]$;

d) $\mu_y[z(x_1, x_2, \ldots, x_n, y) < g(x_1, x_2, \ldots, x_n, y)]$;

e) $\mu_y[z(x_1, x_2, \ldots, x_n, y) = 0 \;\&\; g(x_1, x_2, \ldots, x_n, y) = 0]$;

f) $\mu_y[z(x_1, x_2, \ldots, x_n, y) = 0 \vee g(x_1, x_2, \ldots, x_n, y) = 0]$;

g) $\mu_y[z(x_1, x_2, \ldots, x_n, y) = 0 \vee g(x_1, x_2, \ldots, x_n, y) \leq h(x_1, x_2, \ldots, x_n, y)]$.

9. Покажіть, що наведені нижче функції обчислюються машинами Поста і машинами Тьюрінга:

a) $o(n)$ і $I_m^k(x_1, x_2,..., x_k)$; b) $m + n$, $m \div n$;

c) $t(n) = 2n$; d) $conv(x_1x_2...x_k) = x_kx_{k-1}...x_1$.

10. Побудуйте машину Тьюрінга, яка обчислює

a) добуток; b) частку від ділення;

c) модуль різниці двох натуральних чисел.

11. Побудуйте машину Тьюрінга, яка моделює роботу заданої машини Поста.

12. Побудуйте машину Поста, яка моделює роботу заданої машини Тьюрінга.

13. Побудуйте машину Тьюрінга, яка виконує конкатенацію двох слів із $F(X)$.

14. Покажіть, що проблема зупинки алгоритмічно розв'язувана для машини Тьюрінга, програма якої складається з однієї команди.

15. Побудуйте машину Тьюрінга, яка стирає з інформаційної стрічки довільне початкове слово і записує на стрічку слово 11001 в алфавіті {0, 1}.

16. Які функції можна одержати з найпростіших функцій за допомогою лише оператора суперпозиції?

17. Довести, що із функцій $o(x)$ і $I_m^n(x_1, ..., x_n)$ за допомогою операторів суперпозиції і примітивної рекурсії неможливо одержати функції $s(x) = x + 1$ і $2x$.

3.6. Алгоритмічна система Маркова

Алгоритмічна система Маркова будується майже за тими самими принципами, що й системи Поста і Тьюрінга, але на відміну від них носить дещо простіший і інтуїтивно більш зрозумілий характер.

3.6.1. Вільні напівгрупи і алгорифми Маркова

Нехай X — деякий скінченний алфавіт, $F(X)$ — напівгрупа слів у цьому алфавіті і $e \in F(X)$, де e — пусте слово [28, 15]. Якщо p, $q \in F(X)$, то вирази $p \to q$ і $p \to .q$ називаються **формулами підстановки** в алфавіті X. При цьому вважається, що символи $\to$ і $\cdot$ не належать алфавіту X, а слова p і q можуть бути пустими.

Формула підстановки $p \to q$ називається *простою*, а формула підстановки $p \to .q$ — *заключною*.

Нехай вираз $p \to [.]q$ означає довільну із формул підстановки $p \to q$ або $p \to .q$. Скінченна послідовність R формул підстановки в алфавіті X

$$\begin{cases} p_1 & \to & [.]q_1 \\ p_2 & \to & [.]q_2 \\ \dots & \dots & \dots \\ p_l & \to & [.]q_l \end{cases}$$

називається *схемою* або *системою переписування*. Довільна система переписування являє собою функцію f: $F(X) \to F(X)$, значення якої обчислюються за такими правилами.

(1) Якщо жодне із слів p_i ($i = 1, 2,..., l$) не є підсловом слова p, то p залишається без змін і є результатом переписування. Цей факт будемо записувати у вигляді R: $p!$.

(2) Якщо серед слів $p_1, p_2, \dots, p_l$ існують такі, що є підсловами слова p, то нехай m таке найменше число, що $1 \le m \le l$ і p_m — підслово слова p. Слово p', яке одержане в результаті підстановки самого лівого входження (першого входження) слова p_m словом q_m у слові p, будемо позначати R: $p \vdash p'$.

Робота за даними правилами може завершитися з двох причин:

а) якщо формула підстановки $p_m \to [.]q_m$ заключна;

б) якщо R: $p \models q$ означає, що існує така послідовність $r_0, r_1,..., r_k$ слів із $F(X)$, що $p = r_0$, $q = r_k$ і $R : r_i \vdash r_{i+1}$ для $i = 1, 2,..., k-2$ і або R: $r_k!$, або $R : r_{k-1} \to .r_k$.

Функція f: $F(X) \to F(X)$, визначена таким чином, називається **нормальним алгорифмом Маркова в алфавіті X.**

Робота алгорифму R може бути описана так. Нехай $p \in F(X)$. Знаходимо в R першу формулу підстановки $p_m \to [.]q_m$, таку що p_m підслово слова p. Виконуємо заміну першого входження слова p_m словом q_m у слові p. Якщо p' — результат цієї підстановки, то якщо $p_m \to [.]q_m$ заключна, то робота алгорифму закінчується і результатом є слово p'. Якщо формула підстановки $p_m \to [.]q_m$ — проста, то до слова p' застосовується той самий пошук, що й до слова p і т. д. Якщо, нарешті, одержується таке слово r_i, що $R : r_i!$, тобто ні одне із слів p_i ($i = 1, 2,..., l$ не є підсловом слова r_i, то робота алгорифму закінчується і r_i буде його значенням.

Вище зазначалося, що можлива ситуація, коли описаний процес ніколи не закінчиться. У цьому випадку будемо говорити, що алгорифм R не є **застосовним** до слова p.

Приклади. 1. Нехай $X = \{a, b\}$, а R має вигляд

$$\begin{cases} a & \to & .e \\ b & \to & b. \end{cases}$$

Алгорифм R переписує довільне слово p із $F(X)$, яке має хоча б одну літеру a, у слово, яке отримується з p шляхом викреслювання першого входження літери a в слово p. Зрозуміло також, що $R(e) = e$ і R не застосовний до непустих слів із $F(X)$, які не мають входження літери a.

2. Нехай $X = \{x, y, x^{-1}, y^{-1}\}$, а R має вигляд

$$\begin{cases} xx^{-1} & \to & e \\ x^{-1}x & \to & e \\ yy^{-1} & \to & e \\ y^{-1}y & \to & e. \end{cases}$$

Алгорифм R переписує довільне слово p із $F(X)$ у слово q, яке не має підслів вигляду xx^{-1}, $x^{-1}x$, yy^{-1}, $y^{-1}y$. Наприклад, слово $xxyxyy^{-1}x^{-1}y^{-1}yx$ переписується таким чином: $xxyxyy^{-1}x^{-1}y^{-1}yx \to xxyxx^{-1}y^{-1}yx \to xxyy^{-1}yx \to xxyx$.

3. Якщо $X = \{x_1, x_2, \ldots, x_n\}$, а система R має вигляд

$$\begin{cases} x_1 & \to & e \\ x_2 & \to & e \\ \ldots & \ldots & \ldots \\ x_n & \to & e, \end{cases}$$

то алгорифм R переписує довільне слово p із $F(X)$ у пусте слово e. ♠

Нехай R і Q — нормальні алгорифми над алфавітом X і $p \in F(X)$. Запис $R(p) \approx Q(p)$ означає, що або обидва алгорифми R і Q не застосовні до слова p, або обидва застосовні і $R(p) = Q(p)$. Два алгорифми R і Q над алфавітом X називаються **повністю еквівалентними**, якщо $(\forall p \in F(X))\ R(p) \approx Q(p)$. Алгоритми R і Q називаються **еквівалентними відносно алфавіту** X, якщо $R(p) \approx Q(p)$ щоразу, коли $p \in F(X)$, і хоча б одне зі слів $R(p)$ або $Q(p)$ визначене і теж належить $F(X)$.

Нехай $X = \{1\}$, а $X' = \{1, *\}$, тоді довільне натуральне число n можна записати у вигляді слова $\overline{n}$ у алфавіті X'. Дійсно, це можна зробити за допомогою такої відповідності:

$$\overline{0} — 1,\ \overline{1} — 11,\ \overline{2} — 111,\ \overline{3} — 1111,\ \overline{4} — 11111, \ldots .$$

Поставимо тепер у відповідність довільному вектору $(n_1, n_2, \ldots, n_k)$, де $n_1, n_2, \ldots, n_k$ — натуральні числа, слово в алфавіті X' вигляду $\overline{n}_1 * \overline{n}_2 * \ldots \overline{n}_k$, яке позначимо $\overline{(n_1, n_2, \ldots, n_k)}$.

Наприклад, вектору $\overline{(4, 0, 2)}$ відповідає слово $11111 * 1 * 111$.

Нехай $f: N^k \to N$ — деяка часткова функція, і R_f означає алгоритм в алфавіті X' такий, що

$$R_f\left(\overline{(k_1, k_2, ..., k_n)}\right) = f(k_1, k_2, ..., k_n)$$

тоді і тільки тоді, коли хоча б одна з частин цієї рівності визначена. При цьому вважається, що R_f не застосовний до слів, відмінних від слів вигляду $\left(\overline{k_1, k_2, ..., k_n}\right)$.

3.6.2. Властивості алгорифмів Маркова

Функцію f будемо називати **частково обчислюваною за Марковим**, якщо існує нормальний алгорифм Q над X' повністю еквівалентний R_f над алфавітом X'. Якщо функція f повністю визначена, то її називають просто **обчислюваною за Марковим.**

Теорема 66. *Найпростіші функції* $o(x) = 0$, $s(x) = x + 1$ *і* $I_m^n(x_1, x_2, \ldots, x_n) = x_m$ *обчислювані за Марковим.*

Доведення зводиться до побудови відповідних нормальних алгорифмів.

а) Неважко переконатися, що функцію $o(\overline{x})$ реалізує такий алгорифм R_0:

$$\begin{cases} * & \to & * \\ \alpha 1\,1 & \to & \alpha 1 \\ \alpha 1 & \to & .1 \\ e & \to & \alpha, \end{cases}$$

що застосовний до всіх слів у алфавіті X', які є натуральними числами, де $\alpha \notin X'$.

Дійсно, нехай $p = 11\ldots11$ — довільне слово в алфавіті X. Тоді відповідно до четвертої формули підстановки $R_0: p \vdash \alpha p$.

Після цього, застосовуючи необхідне число разів другу формулу підстановки, одержуємо R_0: $\alpha p \vdash \alpha 1$ і, нарешті, застосовуючи заключну третю формулу підстановки, маємо R_0: $\alpha 1 \vdash 1$. Оскільки 1 представляє число 0 в алфавіті X, то R_0 і є шуканим алгорифмом в силу довільності слова p.

б) Для функції $s(x)$ таким алгорифмом буде алгорифм R_s вигляду

$$\begin{cases} * & \to & * \\ \alpha 1 & \to & .11 \\ e & \to & \alpha, \end{cases}$$

де $\alpha \notin X'$. Цей алгорифм застосовний до тих і тільки тих слів, які є словами в алфавіті X' і представляють натуральні числа, причому $R_s(\overline{n}) = n+1$ для довільного $n \in N$.

в) Більш складну структуру має алгорифм обчислення функції I_m^n. Нехай $\alpha_1, \alpha_2, \ldots, \alpha_{2n} \notin X'$ і $1 \le m < n$, тоді позначимо Sub_m список формул підстановки

$$\begin{cases} \alpha_{2m-1} * & \to & \alpha_{2m-1} * \\ \alpha_{2m-1} 1 & \to & \alpha_{2m} 1 \\ \alpha_{2m} 1 & \to & \alpha_{2m} \\ \alpha_{2m} * & \to & \alpha_{2m+1}. \end{cases}$$

Тепер нормальний алгорифм, який обчислює функцію I_m^n, буде мати такий вигляд:

$$\begin{cases} & m=1 & \\ \alpha_1 * & \to & \alpha_1 * \\ \alpha_1 1 & \to & \alpha_2 1 \\ \alpha_2 1 & \to & 1\alpha_2 \\ \alpha_2 * & \to & \alpha_3 \\ Sub_2 & & \\ Sub_3 & & \\ \ldots & & \\ Sub_{n-1} & & \\ \alpha_{2n-1} * & \to & \alpha_{2n-1} * \\ \alpha_{2n-1} 1 & \to & \alpha_{2n} 1 \\ \alpha_{2n} 1 & \to & \alpha_{2n} \\ \alpha_{2n} * & \to & \alpha_{2n} * \\ \alpha_{2n} & \to & .e \\ e & \to & \alpha_1. \end{cases} \qquad \begin{cases} & 1<m<n & \\ Sub_1 & & \\ Sub_2 & & \\ \ldots & & \\ Sub_{m-1} & & \\ \alpha_{2m-1} * & \to & \alpha_{2m-1} * \\ \alpha_{2m-1} 1 & \to & \alpha_{2m} 1 \\ \alpha_{2m} 1 & \to & 1\alpha_{2m} \\ \alpha_{2m} * & \to & \alpha_{2m+1} \\ Sub_{m+1} & & \\ \ldots & & \\ Sub_{n-1} & & \\ \alpha_{2n-1} * & \to & \alpha_{2n-1} * \\ \alpha_{2n-1} 1 & \to & \alpha_{2n} 1 \\ \alpha_{2n} 1 & \to & \alpha_{2n} \\ \alpha_{2n} * & \to & \alpha_{2n} * \\ \alpha_{2n} & \to & .e \\ e & \to & \alpha_1. \end{cases}$$

$$
\begin{cases}
 & m = n & \\
Sub_1 & & \\
Sub_2 & & \\
\dots & & \\
Sub_{n-1} & & \\
\alpha_{2n-1} * & \to & \alpha_{2n-1} * \\
\alpha_{2n-1} 1 & \to & \alpha_{2n} 1 \\
\alpha_{2n} 1 & \to & 1 \alpha_{2n} \\
\alpha_{2n} * & \to & \alpha_{2n} * \\
\alpha_{2n} & \to & .e \\
e & \to & \alpha_1 .
\end{cases}
$$

Дійсно, після першого застосування алгорифму до слова $x_1 * x_2 * * \dots * x_n$ матимемо слово $\alpha_1 x_1 * x_2 * \dots * x_n$. Після цього Sub_1, Sub_2, ..., Sub_{m-1}, Sub_{m+1},..., Sub_n викреслюють усі x_i від 1 до $m - 1$ і від $m + 1$ до n включно, залишаючи слово x_m. Подробиці рекомендується розглянути самостійно як вправу. ■

Нормальний алгорифм називається **замкнутим**, якщо його схема включає формулу підстановки вигляду $e \to .q$. При роботі такого алгорифму зупинка можлива лише в тому випадку, коли застосовується заключна формула підстановки. Довільний нормальний алгорифм можна перетворити до замкнутого алгорифму шляхом добавки до його схеми в кінці нової формули підстановки вигляду $e \to .e$. Якщо позначити Г· схему одержаного алгорифму, то очевидно, що Г· замкнутий і повністю еквівалентний алгорифму Г.

Теорема 67. *Композиція двох нормальних алгорифмів є нормальним алгорифмом.*

Доведення. Нехай Г, Δ — нормальні алгорифми в алфавіті A. Співставимо кожному символу b цього алфавіту символ b^d, який назвемо двійником символу b. Нехай A^d — алфавіт двійників символів алфавіту A. Виберемо ще будь-які два символи α і β, які не належать $A \cup A^d$. Позначимо Σ_Γ, схему алгорифму, яка виходить із схеми нормального алгорифму Г·, заміною в ній точки в кожній заключній формулі підстановки символом α, і позначимо через $\Sigma_{\overline{\Delta}}$ схему, яка одержана шляхом заміни в схемі алгорифму Δ· всіх символів алфавіту A їх двійниками, усіх точок — символами β з подальшою заміною всіх формул підстановки вигляду $e \to Q$ і $e \to .Q$ відповідно формулами підстановки $\alpha \to \alpha Q$ і $\alpha \to \alpha\beta Q$. Розглянемо схему (у скороченому вигляді):

$$
\begin{cases}
a\alpha \to \alpha a & (a \in A) \\
\alpha a \to \alpha a^d & (a \in A) \\
a^d b \to a^d b^d & (a, b \in A) \\
a^d \beta \to \beta a^d & (a \in A) \\
\beta a^d \to \beta a & (a \in A) \\
a b^d \to a b & (a \in A) \\
\alpha\beta \to .e & \\
\Sigma_{\overline{\Delta}} & \\
\Sigma_{\Gamma}.
\end{cases}
$$

Нормальний алгорифм Θ над алфавітом A, що визначається цією схемою, такий що для довільного слова $p \in A$ $\Theta(p) \approx \Delta(\Gamma(p))$ (див. вправу 5 у кінці параграфа). Цей нормальний алгорифм називається **композицією алгорифмів** Γ і Δ і позначається також символом $\Delta \cdot \Gamma$. У загальному випадку запис $\Gamma_n \cdot \Gamma_{n-1} \cdot \ldots \cdot \Gamma_1$ буде означати $\Gamma_n(\Gamma_{n-1}(\ldots(\Gamma_2 \cdot \Gamma_1)\ldots))$.

Нехай Δ — деякий нормальний алгорифм в алфавіті A і B — деяке розширення алфавіту A. До схеми алгорифму Δ добавимо зверху всі можливі формули підстановки вигляду $b \to b$, де b — довільний символ із $B \setminus A$. Одержана таким чином схема визначає деякий нормальний алгорифм Δ_B в алфавіті B, який не застосовний ні до якого слова, що включає символи із алфавіту $B \setminus A$, і такий, що $\Delta_B(p) = \Delta(p)$ для довільного слова $p \in A$. Алгорифм Δ_B повністю еквівалентний алгорифму Δ відносно алфавіту A і називається **формальним розповсюдженням** алгорифму Δ на алфавіт B.

Нехай дано нормальні алгорифми Γ і Δ відповідно в алфавітах A_1 і A_2. Розглянемо алфавіт $A = A_1 \cup A_2$ і формальні розповсюдження Γ_A і Δ_A алгорифмів Γ і Δ на алфавіт A. Композиція Θ алгорифмів Γ_A і Δ_A називається **нормальною композицією алгорифмів** Γ і Δ і позначається $\Delta \cdot \Gamma$. (Непорозумінь з повторним введенням символу $\Delta \cdot \Gamma$ не виникає, оскільки у випадку, коли $A_1 = A_2$, нормальна композиція алгорифмів $\Delta \cdot \Gamma$ збігається з їх композицією). Θ є нормальним алгорифмом над A_1, причому $\Theta(p) \approx \Delta(\Gamma(p))$ для довільного слова p в A_1, і крім того, Θ застосовний тільки до тих слів p в алфавіті A, які задовольняють умовам:

(i) p є слово в алфавіті A_1;

(ii) Γ застосовний до p;

(iii) Δ застосовний до $\Gamma(p)$.

Припустимо, що алфавіт B є розширенням алфавіту A, і нехай p — довільне слово в алфавіті B. Проекцією p^A слова p на алфавіт A нази-

вається слово, одержане із слова p, коли в p стерти всі входження символів із $B \setminus A$. Скорочено записана схема

$$\{x \to e \quad (x \in B \setminus A)$$

задає нормальний алгорифм $\Delta_{B,A}$ такий, що $\Delta_{B,A}(p) = p^A$ для довільного слова p у B. Алгорифм $\Delta_{B,A}$ називається **алгорифмом-проекцією.**

Нехай A і C — алфавіти без спільних символів. Нехай $B = A \cup C$. Тоді скорочено записана схема

$$\{ca \to ac \quad (a \in A,\ c \in C)$$

задає нормальний алгорифм $\Pi_{A,C}$ в алфавіті B такий, що $\Pi_{A,C}(p) = p^A p^C$ для довільного слова p у B.

Якщо Г — нормальний алгорифм в алфавіті A і B — розширення A, то нормальний алгорифм Δ в алфавіті B, який задається схемою алгорифму Г, називається **звичайним розповсюдженням алгорифму** Г на алфавіт B. Очевидно, що $\Delta(p) \approx \Gamma(p)$ для довільного слова p в A і, крім того, $\Delta(pq) \approx \Gamma(p)q$ для довільного слова p в A і для довільного слова q в $B \setminus A$. Зауважимо, що звичайне розповсюдження алгорифму Г на B відрізняється від формального розповсюдження Г на B, оскільки формальне розповсюдження не застосовне ні до якого слова, що включає символи із $B \setminus A$.

Теорема 68. *Нехай $\Gamma_1,..., \Gamma_k$ — нормальні алгорифми і A — об'єднання їх алфавітів. Тоді існує нормальний алгорифм Δ над A, який називається* **з'єднанням алгорифмів** *$\Gamma_1,..., \Gamma_k$ такий, що $\Delta(p) \approx \approx \Gamma_1^{\#}(p)\Gamma_2^{\#}(p)...\Gamma_k^{\#}(p)$ для довільного слова p в алфавіті A, де $\Gamma_i^{\#}$ — звичайне розповсюдження Γ_i на A.*

Доведення. Покажемо спочатку, що дана теорема має місце для $k = 2$, а потім за допомогою індукції за числом k можна одержати і загальний випадок. Введемо алфавіт двійників A^d для символів алфавіту A. Припустимо $B = A \cup A^d$. Нехай $\overline{\Gamma}_1$ — нормальний алгорифм, схема якого виходить шляхом заміни кожного символу схеми алгорифму Γ_1 її двійником, і нехай $\overline{\Gamma}_1^{\#}$, $\Gamma_2^{\#}$ — звичайні розповсюдження відповідно алгорифмів $\overline{\Gamma}_1$ і Γ_2 на B. Нехай $A = \{a_1,..., a_n\}$ і $\Sigma = Sub(a_1,..., a_n : q_1,..., q_n)$ — нормальний алгорифм, який задається схемою

$$\begin{cases} \alpha a_i & \to & q_i\alpha & (i = 1, 2,..., n) \\ \alpha\xi & \to & \xi\alpha & (\xi \in A \setminus \{a_1,..., a_m\}) \\ \alpha & \to & .e & \\ e & \to & \alpha & \end{cases}$$

і виконує одночасну підстановку слів $q_1,..., q_n$ у слово p відповідно замість символів $a_1,...,a_n$. Тоді нормальні алгорифми $\Sigma_1 = Sub(a_1,..., a_n : a_1a_1^d,..., a_na_n^d)$ і $\Sigma_2 = Sub(a_1^d,...,a_n^d : a_1,..., a_n)$ над B, такі, що Σ_1 виконує одночасну підстановку $a_1^d,...,a_n^d$ замість $a_1,..., a_n$, а Σ_2 — одночасну підстановку $a_1,..., a_n$ замість $a_1^d,...,a_n^d$. Існують, крім того, нормальні алгорифми Ξ_{A,A^d} і $\Xi_{A^d,A}$ такі, що $\Xi_{A,A^d}(p) = p^A p^{A^d}$, $\Xi_{A^d,A}(p) = p^{A^d} p^A$. Тоді, як неважко перевірити, нормальна композиція $\Delta = \Sigma_2 \cdot \overline{\Gamma}_1^{\#} \cdot \Xi_{A^d,A} \cdot \Gamma_2^{\#} \cdot \Xi_{A,A^d} \cdot \Sigma_1$ має шукану властивість: $\Delta(p) = \Gamma_1^{\#}(p) \cdot \Gamma_2^{\#}(p)$ для довільного слова p в алфавіті A. ■

Наслідок 22. *Нехай $\Gamma_1,..., \Gamma_k$ — нормальні алгорифми відповідно над алфавітами $A_1,..., A_k$, і нехай $A = A_1 \cup ... \cup A_k$. Тоді існує нормальний алгорифм Δ над $A \cup \{*\}$ такий, що $\Delta(p) = \Gamma_1^{\#}(p) * \Gamma_2^{\#}(p) * ... \Gamma_k^{\#}(p)$ для довільного слова p в A, де $\Gamma_i^{\#}$ — звичайне розповсюдження Γ_i на A. Зокрема, $\Delta(p) = \Gamma_1^{\#}(p) * \Gamma_2^{\#}(p) * ... \Gamma_k^{\#}(p)$ для довільного слова p в алфавіті $A_1 \cap ... \cap A_k$.*

Доведення. Існує такий нормальний алгорифм Θ в $A \cup \{*\}$, що $\Theta(p) = *$ для довільного слова p в A. Алгорифм Θ задається схемою

$$\begin{cases} a \to e & (a \in A) \\ e \to .* \end{cases}$$

Тоді, на основі попередньої теореми, Δ є з'єднанням алгорифмів Δ_1, Θ, Δ_2, Θ,..., Θ, Δ_k. Легко помітити, що $\Delta(p) \approx \Gamma_1^{\#}(p) * \Gamma_2^{\#}(p) * ... \Gamma_k^{\#}(p)$ для довільного слова p в A і, зокрема, $\Delta(p) \approx \Gamma_1^{\#}(p) * \Gamma_2^{\#}(p) * ... \Gamma_k^{\#}(p)$ для довільного слова p в алфавіті $A_1 \cap ... \cap A_k$.

Теорема 69. (1) *Нехай Ξ — нормальний алгорифм в алфавіті A і α — довільний символ. Тоді існує нормальний алгорифм Θ над $A \cup \{\alpha\}$ такий, що для довільного слова p в A*

$$\Theta(p) = \begin{cases} \alpha p, & \text{якщо } \Xi(p) = e, \\ p, & \text{якщо } \Xi(p) \neq e \end{cases}$$

і алгорифм Θ застосовний тільки до тих слів, до яких застосовний Ξ.

(2) *Якщо Δ і Γ — нормальні алгорифми в алфавіті A і α — символ, що не належить A, то існує нормальний алгорифм Ψ над $A \cup \{\alpha\}$ такий, що $\Psi(p) \approx \Delta(p)$ і $\Psi(\alpha p) \approx \Gamma(p)$ для довільного слова p в A.*

Доведення. (1) Існує нормальний алгорифм Δ_1 над $A \cup \{\alpha\}$, який переробляє пусте слово e в α і довільне непусте слово в алфавіті $A \cup \{\alpha\}$ — в пусте слово e. Такий алгорифм можна задати, наприклад, схемою:

$$\Theta(p) = \begin{cases} a & \to & \beta & (a \in A \cup \{\alpha\}) \\ \beta\beta & \to & \beta & \\ \beta & \to & .e & \\ e & \to & .\alpha & \end{cases}$$

де β — символ, який не належить алфавіту $A \cup \{\alpha\}$.

Нехай $\Delta_2 = \Delta_1 \cdot \Xi$. Для довільного слова p в A, якщо $\Xi(p) = e$, то $\Delta_2(p) = \alpha$, і якщо $\Xi(p) \neq e$, то $\Delta_2(p) = e$. Нехай T — тотожний нормальний алгорифм (схема якого має вигляд $\{e \to .e\}$) і Θ — з'єднання алгорифмів Δ_2 і T. Тоді якщо $\Xi(p) = e$, то $\Theta(p) = \alpha p$, і якщо $\Theta(p) \neq e$, то $\Theta(p) = p$.

(2) Введемо алфавіт A^d двійників символів алфавіту A. Нехай $B = A \cup A^d \cup \{\alpha, \beta\}$, де $\beta \notin A \cup A^d \cup \{\alpha\}$. Якщо замінити в схемі алгорифму $\Delta\cdot$ довільну літеру алфавіту A її двійником, усі точки — літерою β і в одержаній схемі замінити довільну формулу підстановки вигляду $e \to q$ і $e \to .q$ відповідно на $\alpha \to \alpha q$ і $\alpha \to \alpha\beta q$, то одержимо деяку схему, яку позначимо $\Sigma_{\bar{\Delta}\cdot}$. Нехай $\Sigma_{\Gamma\cdot}$ — схема алгорифму $\Gamma\cdot$. Побудуємо схему

$$\begin{cases} \alpha a & \to & \alpha a^d & (a \in A) \\ a^d b & \to & a^d b^d & (a, b \in A) \\ a^d \beta & \to & \beta a^d & (a \in A) \\ \beta a^d & \to & \beta a & (a \in A) \\ a b^d & \to & a b & (a, b \in A) \\ \alpha\beta & \to & .e & \\ \Sigma_{\bar{\Delta}\cdot} & & & \\ \Sigma_{\Gamma\cdot} & & & \end{cases}$$

Нормальний алгорифм Ψ, який задається цією схемою над $A \cup \{\alpha\}$, є шуканим алгорифмом, тобто $\Psi(p) \approx \Delta(p)$ і $\Psi(\alpha p) \approx \Delta(p)$ для довільного слова p в A.

Теорема 70. *Нехай* Γ, Δ, Ξ — *нормальні алгорифми і* A — *об'єднання їх алфавітів. Тоді існує нормальний алгорифм* Ψ *над* A *такий, що*

$$\Psi(p) = \begin{cases} \Delta(p), & \text{якщо } p \text{ — слово в алфавіті } A \text{ і } \Xi(p) = e, \\ \Gamma(p), & \text{якщо } p \text{ — слово в алфавіті } A \text{ і } \Xi(p) \neq e, \end{cases}$$

і який застосовний до тих і тільки тих слів у A, до яких застосовний Ξ. Алгорифм Ψ називається **розгалуженням алгорифмів** Γ *і* Δ, *під керівництвом алгорифму* Ξ.

Доведення. Нехай Γ_1, Δ_1, Ξ_1 — формальні розповсюдження відповідно алгорифмів Γ, Δ, Ξ на A і α — символ, що не належить до A. За теоремою 69 (1) існує такий нормальний алгорифм Θ над $A \cup \{\alpha\}$, що

$$\Theta(p) = \begin{cases} \alpha p, & \text{якщо } p \text{ — слово в алфавіті } A \text{ і } \Xi(p) = e, \\ p, & \text{якщо } p \text{ — слово в алфавіті } A \text{ і } \Xi(p) \neq e. \end{cases}$$

Крім того, за теоремою 69 (2), існує такий нормальний алгорифм Φ над $A \cup \{\alpha\}$, що коли p — слово в алфавіті A, то $\Phi(p) \approx \Gamma_1(p)$ і $\Phi(\alpha p) \approx \Delta_1(p)$. Тепер залишається припустити $\Psi = \Phi \cdot \Theta$. ■

Нехай дано алгорифми Γ і Ξ в алфавіті A і довільне слово p_0 в A. Застосуємо Γ до p_0. Якщо в результаті вийде деяке слово p_1, то застосуємо Ξ до p_1. Якщо виявиться, що $\Xi(p_1) = e$, то процес закінчується. Якщо $\Xi(p_1) \neq e$, то знову застосуємо Γ до p_1. Коли в результаті цього вийде деяке слово p_2, то застосуємо Ξ до p_2, і знову, якщо виявиться, що $\Xi(p_2) = e$, то процес зупиняється, а якщо $\Xi(p_2) \neq e$, то застосуємо Γ до p_2 і т. д. Визначений таким чином алгорифм Δ називається **повторенням алгорифму** Γ, що управляється алгорифмом Ξ. Очевидно, що $\Delta(p_0) = q$ тоді і тільки тоді, коли існує послідовність слів $p_0, p_1, \ldots, p_n$ $(n > 0)$ така, що $p_n = q$, $\Xi(p_n) = e$, $p_i = \Gamma(p_{i-1})$ при $0 < i \leq n$ і $\Xi(p_i) \neq e$ при $0 < i < n$.

Теорема 71. *Нехай* Γ і Ξ — *нормальні алгорифми,* A — *об'єднання їх алфавітів і* Γ_1 *і* Ξ_1 — *формальні розповсюдження відповідно* Γ *і* Ξ *на* A. *Тоді існує нормальний алгорифм* Δ *над* A, *який є повторенням алгорифму* Γ_1, *що управляється алгорифмом* Ξ_1.

Доведення. Теорему, очевидно, достатньо довести для випадку, коли алфавіти алгорифмів Γ і Ξ збігаються, і тоді $\Gamma_1 = \Gamma$ і $\Xi_1 = \Xi$. Нехай літера α не належить до A. За теоремою 69 (1) існує такий нормальний алгорифм Θ над $B = A \cup \{\alpha\}$, що

$$\Theta(p) = \begin{cases} \alpha p, & \text{якщо } p \text{ — слово в алфавіті } A \text{ і } \Xi(p) = e, \\ p, & \text{якщо } p \text{ — слово в алфавіті } A \text{ і } \Xi(p) \neq e. \end{cases}$$

Нехай $T = \Theta \cdot \Gamma$. T — нормальний алгорифм у деякому розширенні F алфавіту B. Нехай літера β не належить до F. Розглянемо таку схему:

$$\begin{cases} \xi\beta \to \beta\xi & (\xi \in F) \\ \beta\alpha \to .\alpha \\ \beta \to e \\ \Sigma_T^{\beta} \end{cases}$$

де Σ_T^{β} — схема, яка одержана зі схеми T шляхом заміни в ній усіх крапок літерою β. Ця схема задає деякий нормальний алгорифм Φ, такий що $\Phi(p) = q$ тоді і тільки тоді, коли існує послідовність слів p_0, p_1,..., p_n така, що $p_0 = p$, $p_n = q$, $p_i = T(p_{i-1})$ $(0 < i \le n)$ і p_n єдине в цій послідовності слово, що починається з літери α. Нехай Π — алгорифм, який проектує алфавіт F на алфавіт $F \setminus \{\alpha\}$ (тобто, алгорифм, що витирає всі входження букви α). Тепер легко переконатися, що нормальний алгорифм $\Delta = \Pi \cdot \Phi$ — шуканий. ■

Наслідок 23. *Нехай* Γ *і* Δ *— нормальні алгорифми і* A *— об'єднання їх алфавітів. Тоді існує нормальний алгорифм* Θ *над* A*, який довільне слово* p *в алфавіті* A *переробляє у слово* q *тоді і тільки тоді, коли існує така послідовність слів* p_0,..., p_n $(n \ge 0)$*, що* $p_0 = p$, $p_n = q$, $\Delta(p_n) = e$, $p_{i+1} = \Gamma(p_i)$ *і* $\Delta(p_i) \ne e$ *для* $i = 0, 1,..., n-1$.

Доведення. Нехай T — тотожний нормальний алгорифм і Δ_1 — повторення алгорифму Γ, під управлінням Δ. Шуканим алгорифмом Θ тоді є розгалуження Δ_1 і T, під керівництвом Δ (див. теорему 70). Цей алгорифм Θ називається **повним повторенням алгорифму** Γ, під керівництвом алгорифму Δ.

Теорема 72. *Який би не був нормальний алгорифм* Γ *в алфавіті* A*, існує такий нормальний алгорифм* Γ' *над алфавітом* $B = A \cup C$*, де* $C = \{1, *\}$*, що для довільного слова* p *в* A *і довільного натурального числа* n $\Gamma'(\overline{n} * p) = Q$ *тоді і тільки тоді, коли існує послідовність слів* $p_0, p_1, \ldots, p_n$ $(n \ge 0)$*, що задовольняє умови* $p_0 = p$, $p_n = Q$, $p_i = \Delta(p_{i-1})$*, для* $i = 1, 2,..., n$.

Доведення. Нехай α — будь-яка літера, що не входить до алфавіту B, і нехай $D = B \cup \{\alpha\}$. Розглянемо нормальні алгорифми в D, які задаються такими схемами.

$$\Delta_1 = \begin{cases} \alpha 1\, 1 & \to & .1 \\ \alpha 1 * & \to & \alpha * \\ \alpha * \xi & \to & \alpha * \quad (\xi \in B) \\ \alpha * & \to & .e \\ e & \to & \alpha. \end{cases}$$

Неважко переконатися, що $\Delta_1(\overline{0} * p) = e$ і $\Delta_1(\overline{n} * p) \ne e$, якщо $n > 0$.

$$\Delta_2 = \begin{cases} *\xi & \to & * \quad (\xi \in B) \\ & \to & e. \end{cases}$$

Якщо p не включає символ $*$, то $\Delta_2(p * Q) = p$.

$$\Delta_3 = \begin{cases} \alpha 1 & \to \alpha \\ \alpha * & \to .e \\ e & \to \alpha. \end{cases}$$

Очевидно, що $\Delta_3(\overline{n} * p) = p$ для довільного $p \in F(B)$.

$$\Delta_4 = \{1 \to .e$$
$$\Delta_5 = \{1* \to .\text{e}.$$

Очевидно, що $\Delta_4(\overline{n} * p) = \overline{n-1} * p$, коли $n > 0$ і $\Delta_4(\overline{0} * p) = *p$. Крім того, $\Delta_5(\overline{0} * p) = p$.

Нехай тепер Θ є такий нормальний алгорифм, що

$$\Theta(p) = (\Delta_2 \cdot \Delta_4)(p) * (\Gamma \cdot \Delta_3)(p)$$

для довільного слова $p \in F(D)$ (див. наслідок 22). Звідси одержуємо $\forall p \in F(A)$:

$$\Theta(\overline{n} * p) = \begin{cases} \overline{n-1} * \Gamma(p), & \text{якщо } n > 0, \\ \Gamma(p), & \text{якщо } n = 0. \end{cases}$$

Позначимо через E алфавіт алгорифму Θ. В силу наслідка 23 знайдеться такий нормальний алгорифм Φ над E, що $\Phi(p) = Q$ тоді і тільки тоді, коли існує послідовність слів $p_0, p_1, \ldots, p_n$ $(n \geq 0)$, що задовольняє умови: $p_0 = p$, $p_n = Q$, $\Delta_1(p_n) = e$, $p_i = \Theta(p_{i-1})$, $(0 < i \leq n)$ і $\Delta_1(p_i) \neq e$ $(0 \leq i < n)$. Тепер неважко переконатися, що $\Gamma' = \Delta_5 \cdot \Phi$. ■

Теорема 73. *Довільна частково рекурсивна функція є частково обчислюваною за Марковим.*

Доведення. Для того щоб довести теорему, необхідно показати обчислюваність за Марковим найпростіших функцій, а також обчислюваність операцій суперпозиції, примітивної рекурсії і мінімізації частково рекурсивних функцій.

(а) Обчислюваність за Марковим *найпростіших функцій* o, s і I_m^n випливає з теореми 66.

(б) *Операція суперпозиції.* Нехай функція φ побудована за функціями $f, f_1, \ldots, f_k$ за допомогою операції суперпозиції:

$$\varphi(x_1, \ldots, x_n) = f(f_1(x_1, \ldots, x_n), \ldots, f_k(x_1, \ldots, x_n)),$$

де $f, f_1, \ldots, f_k$ — частково рекурсивні функції. Припустимо, що існують нормальні алгорифми $\Gamma_f, \Gamma_{f1}, \ldots, \Gamma_{fk}$ над $X' = \{1, *\}$, які частково обчислюють відповідні функції. Тоді, за наслідком 22, існує нормальний алгорифм Δ над X', такий що

$$\Delta(p) \approx \Gamma_{f1}(p) * \Gamma_{f2}(p) * \ldots * \Gamma_{fk}(p)$$

для довільного $p \in F(X')$. Зокрема,

$$\Delta\left(\overline{(x_1,\ldots,x_n)}\right) = \overline{f_1(x_1,\ldots,x_n)} * \ldots * \overline{f_k(x_1,\ldots,x_n)}$$

для довільних натуральних чисел $x_1,\ldots, x_n$. Припустимо $\Theta = \Gamma_f \cdot \Delta$. Тоді для довільних натуральних чисел $x_1,\ldots, x_n$ будемо мати

$$\Theta\left(\overline{(x_1,\ldots,x_n)}\right) = \Gamma_f\left(\overline{f_1(x_1,\ldots,x_n)},\ldots,\overline{f_k(x_1,\ldots,x_n)}\right) =$$
$$= \overline{f(f_1(x_1,\ldots,x_n),\ldots,f_k(x_1,\ldots,x_n))}.$$

(в) *Операція рекурсії.* Нехай функція f побудована за функціями g і h за допомогою операції примітивної рекурсії

$$f(x_1,\ldots, x_n, 0) = g(x_1,\ldots, x_n),$$
$$f(x_1,\ldots, x_n, y+1) = h(x_1, \ldots, x_n, y, f(x_1,\ldots, x_n, y))$$

і нехай Γ_g і Γ_h — нормальні алгорифми над X', які частково обчислюють відповідно функції g і h. Нарешті, нехай Γ_o, Γ_s і Γ_m^n — нормальні алгорифми, які обчислюють найпростіші функції o, s і I_m^n. За допомогою алгорифмів Γ_m^{n+1} можна побудувати (за наслідком 22) такий нормальний алгорифм Δ_1 над X', що

$$\Delta_1(\overline{x}_1 * \ldots * \overline{x}_n * \overline{y}) = \overline{x}_1 * \ldots * \overline{x}_n.$$

Нехай $\Delta = \Gamma_g \cdot \Delta_1$. Виходячи з алгорифмів Γ_{k+1}^{k+1}, Γ_1^{k+1},..., Γ_k^{k+1}, Γ_o, Δ і наслідку 22, будуємо нормальний алгорифм Δ_2 над X', такий що

$$\Delta_2(\overline{x}_1 * \ldots * \overline{x}_n * \overline{y}) = \overline{y} * \overline{x}_1 * \ldots * \overline{x}_n * \overline{0} * \overline{g(x_1,\ldots,x_n)}.$$

Нормальний алгорифм $\Delta_3 = \Gamma_s \cdot \Gamma_{k+1}^{k+2}$ працює так, що $\Delta_3(\overline{x}_1 * \ldots * \overline{x}_n * \overline{y} * \overline{x}) = \overline{y+1}$. Знову, застосовуючи наслідок 22 до алгорифмів $\Gamma_1^{k+2},\ldots,\Gamma_k^{k+2}$, Δ_3 і Γ_f, одержуємо нормальний алгорифм Δ_4 над X' такий, що

$$\Delta_4(\overline{x}_1 * \ldots * \overline{x}_n * \overline{y} * \overline{x}) = \overline{x}_1 * \ldots * \overline{x}_n * \overline{y+1} * \overline{h(x_1,\ldots,x_n,y,x)}.$$

В силу теореми 72 існує такий нормальний алгорифм Δ_4^I, що при довільному $n \geq 0$ рівність $\Delta_4^I(\overline{n} * p) = Q$ виконується тоді і тільки тоді, коли існує послідовність слів $p_0, p_1, \ldots, p_n$, що задовольняє умови $p_0 = p$, $p_n = Q$ і $p_i = \Delta_4(p_{i-1})$, де $0 < i \leq n$. Тоді алгорифм

$$\Delta = \Gamma_{k+2}^{k+2} \cdot \Delta_4^I \cdot \Delta_2$$

і є шуканим алгорифмом, який обчислює функцію f. Дійсно,

$$\Delta_2(\overline{x}_1 * \ldots * \overline{x}_n * \overline{y}) = \overline{y} * \overline{x}_1 * \ldots * \overline{x}_n * \overline{0} * \overline{g(x_1 * \ldots * x_n)}.$$

Застосування Δ_4' до слова $\overline{y} * \overline{x}_1 * \ldots * \overline{x}_n * \overline{0} * \overline{g(x_1 * \ldots * x_n)}$, очевидно, рівнозначне y-кратному застосуванню Δ_4, починаючи зі слова $\overline{x}_1 * \ldots * \overline{x}_n * \overline{0} * \overline{f(x_1 * \ldots * x_n * y)}$. Неважко зрозуміти, що при цьому одержуємо слово $\overline{x}_1 * \ldots * \overline{x}_n * \overline{y} * \overline{f(x_1 * \ldots * x_n * y)}$. Застосувавши до цього слова алгорифм Γ_{k+2}^{k+2}, остаточно одержуємо $\overline{f(x_1 * \ldots * x_n * y)}$.

(г) *Операція мінімізації*. Нехай

$$f(x_1, \ldots, x_n) = \mu_y(g(x_1, \ldots, x_n, y) = 0)$$

і Γ_g — нормальний алгорифм над X', який обчислює функцію g. Виходячи з алгорифмів $\Gamma_1^{n+1}, \ldots, \Gamma_n^{n+1}$ і $\Gamma_s, \ldots, \Gamma_{n+1}^{n+1}$, побудуємо алгорифм M такий, що $M(\overline{x}_1 * \ldots * \overline{x}_n * \overline{y}) = \overline{x}_1 * \ldots * \overline{x}_n * \overline{y+1}$. Розглянемо нормальний алгорифм Θ над X', який визначається схемою

$$\begin{cases} 11 & \to \quad .11 \\ 1 & \to \quad e. \end{cases}$$

Якщо $n = 0$, то $\Theta(\overline{n}) = e$, а якщо $n > 0$, то $\Theta(\overline{n}) \neq e$. Нехай $\Sigma = \Theta \cdot \Gamma_g$, тоді

$$\Sigma(\overline{x}_1 * \ldots * \overline{x}_n * \overline{y}) \begin{cases} = e, & \text{якщо } g(x_1, \ldots, x_n, y) = 0 \\ \neq e, & \text{якщо } g(x_1, \ldots, x_n, y) \neq 0. \end{cases}$$

Нехай тепер Π — нормальний алгорифм над X', такий, що

$$\Pi(\overline{x}_1 * \ldots * \overline{x}_n * \overline{y}) = \overline{x}_1 * \ldots * \overline{x}_n * \overline{0}.$$

Застосовуючи наслідок 23 до алгорифмів M і Σ, матимемо нормальний алгорифм Δ над X', для якого $\Delta(p) = Q$ виконується тоді і тільки тоді, коли існує послідовність $p_0, p_1, \ldots, p_n$, така, що $p_0 = p$, $p_n = Q$, $\Sigma(p_n) = e$, $p_{i+1} = M(p_i)$ і $\Sigma(p_i) \neq e$, для $i = 0, \ldots, n-1$. Визначимо тепер нормальний алгорифм $\Upsilon = \Gamma_{n+1}^{n+1} \cdot \Delta \cdot \Pi$. Неважко бачити, що

$$\Upsilon(\overline{x}_1 * \ldots * \overline{x}_n) = \overline{\mu_y(g(x_1 * \ldots * x_n * y) = 0)} = \overline{f(x_1 * \ldots * x_n)}.$$

Отже, якщо $f(x_1, \ldots, x_n)$ частково рекурсивна функція, то існує деякий нормальний алгорифм, такий, що

$$\Gamma_f(\overline{x}_1 * \ldots * \overline{x}_n) = \overline{f(x_1 * \ldots * x_n)}.$$

Нехай $\Gamma(\overline{x}_1 * ... * \overline{x}_n) = \Gamma_1^n(\overline{x}_1 * ... * \overline{x}_n) * ... * \Gamma_n^n(\overline{x}_1 * ... * \overline{x}_n) = \overline{x}_1 * ... * \overline{x}_n$ тоді $\Sigma(\overline{x}_1 * ... * \overline{x}_n) = \Gamma_f \cdot \Gamma(x_1 * ... * x_n) = \overline{f(x_1 * ... * x_n)}$..

Причому Σ застосовний до тих і тільки тих слів p із $F(X')$, які мають вигляд $\overline{x}_1 * ... * \overline{x}_n$ і для яких $f(x_1,..., x_n)$ визначена. ■

Має місце і обернена теорема.

Теорема 74. *Довільна частково обчислювана за Марковим функція є частково рекурсивною.*

Доведення цієї теореми виходить за межі даного матеріалу і тому не приводиться. За необхідності, це доведення можна знайти, наприклад, у монографіях [26, 28].

З теорем 73 і 74 випливає, що все, що обчислюється, можна обчислити у вільних моноїдах. Цей факт виявляє важливість поняття вільного моноїда і його роль у теорії алгоритмів.

3.7. Контрольні питання, задачі і вправи

Контрольні питання

1) Що таке формула підстановки?
2) Що називається нормальним алгорифмом Маркова?
3) Що собою являє алгоритмічна система Маркова?
4) Дайте визначення функції, обчислюваної за Марковим.
5) В якому випадку алгорифм Маркова завершується результативно і що є результатом його роботи?
6) Сформулюйте теореми про суперпозицію, розгалуження і повторення нормальних алгорифмів.

Задачі і вправи

1. Нехай $X = \{a_1, a_2, ..., a_n\}$ і $p \in F(X)$ — довільне слово. Опишіть дії нормальних алгорифмів Маркова, які задаються схемами:

a) $S = \{e \to .p$

b) $S = \begin{cases} \alpha\xi & \to & \xi\alpha \\ \alpha & \to & .p \\ e & \to & \alpha, \end{cases}$

де $\alpha \in X$, $\xi \notin X$.

c) $S = \begin{cases} a_1 & \to & e \\ a_2 & \to & e \\ \ldots & \ldots & \ldots \\ a_n & \to & e \\ e & \to & .p. \end{cases}$

2. Побудуйте нормальний алгорифм Маркова для обчислення значень таких селекторних функцій:

а) I_2^3 $\{x_1, x_2, x_3\}$; *b)* $I_3^3(x_1, x_2, x_3)$; *c)* $I_4^4(x_1, x_2, x_3, x_4)$.

3. Нехай $p \in F(X)$ і $p = xy\ldots z$. Визначимо такі функції:

а) $head(p) = x$, b) $tail(p) = y\ldots z$.

Побудуйте нормальні алгорифми, які обчислюють значення операцій конкатенації, *head* і *tail* в алфавіті $X = \{a_1, a_2, \ldots, a_n\}$.

4. Побудуйте нормальні алгорифми, які обчислюють значення операцій додавання, модуля різниці двох чисел, зрізаної різниці, множення на 2 і множення двох натуральних чисел.

5. Побудуйте нормальні алгорифми для таких функцій:

а) $s(o(x))$; b) $o(s(x))$; c) $o(I_1^2(x, y))$; d) $I_1^1(s(x))$.

6. Побудуйте нормальний алгорифм Д, який заключає довільне слово p в дужки, тобто $Д(p) = (p)$.

7. Побудуйте а) формальне розповсюдження алгорифму

$$S = \begin{cases} a & \to & .e \\ b & \to & b \end{cases}$$

в алфавітах $A = \{a, b, c\}$ і $B = \{a, b, c, d\}$.

b) композицію алгорифмів Rb (права дужка, тобто $Rb(p) = p$).) і Lb (ліва дужка, тобто $Lb(p) = (p.$).

8. Побудувати схему нормального алгорифму, який обчислює

а) константу 2;

b) $A(n_1 * n_2 * \ldots * n_k) = n_i + 1$, $i = 1, 2, \ldots, k$;

c) $Rb \cdot Lb$;

d) є чи ні задане слово паліндромом.

9. Побудуйте нормальні алгорифми для обчислення таких функцій:

а) $sg(x) = \begin{cases} 0, & \text{якщо } x = 0, \\ 1, & \text{якщо } x \neq 0. \end{cases}$

b) $\overline{sg}(x) = \begin{cases} 1, & \text{якщо } x = 0, \\ 0, & \text{якщо } x \neq 0. \end{cases}$

c) $\delta(x)=\begin{cases}x-1, & \text{якщо } x>0,\\ 1, & \text{якщо } x=0.\end{cases}$

d) $x\div y=\begin{cases}x-y, & \text{якщо } x>y,\\ 0, & \text{якщо } x\le y.\end{cases}$

e) $|x-y|=(x\div y)+(x\div y)$.

f) $|x-y|=\begin{cases}x-y, & \text{якщо } x>y,\\ y-x, & \text{якщо } x\le y.\end{cases}$

g) $\min(x,y)=x\div(x\div y),\ x>y,\ x\ge y,\ x\ne y.$

$\max(x,y)=\begin{cases}x, & \text{якщо } x>y,\\ y, & \text{якщо } x\le y.\end{cases}$

h) суми перших n членів арифметичної прогресії.

i) $\min(x_1,x_2,...,x_k)=\begin{cases}x_1, & \text{якщо } k=1,\\ \min(\min(x_1,x_2),x_3,...,x_k), & \text{якщо } k>1.\end{cases}$

10. Покажіть, що алгорифм, наведений у теоремі 67, дійсно є суперпозицією алгорифмів Γ і Δ, а алгорифм, наведений у теоремі 69, виконує одночасну підстановку слів $p_1,..., p_k$ замість слів $q_1,..., q_k$ у слові p.

11. Побудуйте композицію таких алгорифмів у алфавіті $A = \{a, b\}$: a) *conc* і *tail*, b) *tail* і *conc* (див. наступний параграф).

12. Побудуйте з'єднання алгорифмів *conc* і *tail* в алфавіті $A = \{a, b\}$.

13. Побудуйте розгалуження алгорифмів *tail* і *conc* в алфавіті $A = \{a, b\}$.

14. Побудуйте повторення алгорифму *tail* під керівництвом алгорифму $o(x)$.

15. Побудуйте нормальний алгорифм A в алфавіті $X' = \{*, 1\}$, такий, що

a) $A(n)=\begin{cases}\alpha n, & \text{якщо } n=0,\\ n, & \text{якщо } n\ne 0.\end{cases}$

b) є повторенням алгорифму $s(x)$ під керівництвом алгорифму $sg(x)$.

3.8. Поняття про функціональні мови. Алгоритмічна повнота функціональних мов

У цьому підрозділі розглядається алгебра і алгебраїчна система списковых структур. Доводиться алгоритмічна повнота цієї системи. Застосовуючи результати попереднього розділу і алгоритмічну повноту алгебраїчної системи списковых структур, показується алгоритмічна повнота функціональної мови програмування ЛІСП.

3.8.1. Алгебра списковых структур

Списки. Операції над списками. Нехай $F(X)$ — вільна напівгрупа з одиницею над деяким скінченним алфавітом $X = \{x_1, x_2,..., x_n\}$. Роль одиниці відіграє пусте слово. Нагадаємо, що словом в алфавіті X називається довільна скінченна послідовність символів цього алфавіту. Довільне слово $p = y_1y_2 \dots y_m$ із $F(X)$ будемо називати **списком** елементів $y_1y_2 \dots y_m$, а самі елементи $y_i \in X$, $i = 1, 2,..., m$, — складовими цього списку. При цьому елемент y_1 називається *початком*, а елемент y_m — *кінцем списку*. Якщо $p \in F(X)$, то число складових списку p називається його довжиною і позначається через $l(p)$. Якщо p, q — два списки, то список (слово) q називається *початком* (*кінцем*) списку (слова) p, коли існує таке слово p', що $p = qp'$ ($p = p'q$). Два списки $p = s_1s_2...s_k$ і $q = t_1t_2...t_l$ рівні між собою, якщо $l = k$ (тобто $l(p) = l(q)$) і $s_i = t_i$, $i = 1, 2,..., k$.

З теорії відомо, що $F(X)$ є алгеброю з однією бінарною операцією конкатенації (*conc*) і однією нульарною операцією (пусте слово e). Введемо в розгляд ще декілька функцій і операцій над списками, тобто над елементами множини $F(X)$ [15].

Нехай N — множина натуральних чисел і $p = y_1y_2 \dots y_m$ — довільне слово із $F(X)$, тоді

(1) $head(p) = y_1$ ($head$: $F(X) \to F(X)$).

Іншими словами, функція $head(p)$ дає перший символ слова p. Безпосередньо з визначення цієї функції випливають такі її властивості:

$head(e) = e$, $head(y) = y$, якщо $y \in X$, $head(head(p)) = head(p)$.

(2) $tail(p) = y_2 \dots y_m$ ($tail$: $F(X) \to F(X)$).

Очевидно, що

$$tail\,(e) = e, \qquad tail\,(y) = e, \text{ якщо } y \in X.$$

Зміст наведених нижче функцій випливає з їх визначення.

(3) $add(p, i, x) = y_1 \dots y_i x y_{i+1} \dots y_m$, $0 \le i \le l(p)$.

(4) $sub(p, i) = y_1 \dots y_{i-1} y_{i+1} \dots y_m$, $1 \le i \le l(p)$.

(5) $dist(p, i) = (p_1, p_2)$, де $p_1 = y_1 \dots y_i$, $p_2 = y_{i+1} \dots y_m$, $0 \le i \le l(p)$.

(6) $hl(p, i) = y_1 \dots y_i$, $0 \le i \le l(p)$.

(7) $tr(p, i) = y_{i+1} \dots y_m$, $0 \le i \le l(p)$.

(8) $push(p, x) = px = add(p, l(p), x)$.

(9) $pop(p) = y_1 \dots y_{m-1} = sub(p, l(p))$.

Виявляється, що базовими операціями, тобто такими, через які виражаються всі останні із перелічених функцій, є операції e, *conc*, *head* і *tail* і рекурсії. Іншими словами має місце таке просте твердження.

Теорема 75. *Усі функції* (3)—(9) *представляються у вигляді термів за допомогою операцій e, conc, head, tail і оператора рекурсії.*

Доведення. Розглянемо послідовно випадки:

(3) $add(p, i, x) = hl(p, i)\, xtr\, (p, i)$.

(4) $sub(p, i) = hl(p, i - 1)\, tr\, (p, i)$.

(5) $dist(p, i) = (hl(p, i), tr(p, i))$.

(6) $$hl(p,i) = \begin{cases} e, & \text{якщо } i = 0, \\ head(p) & \text{якщо } i = 1, \\ head(p)\, hl(tail(p), i-1), & \text{якщо } i > 1, \end{cases}$$

(7) $$tr(p,i) = \begin{cases} p, & \text{якщо } i = 0, \\ tail(p) & \text{якщо } i = 1, \\ tr(tail(p), i-1), & \text{якщо } i > 1, \end{cases}$$

З представлення функцій *hl* і *tr* випливає представлення функцій *add*, *sub* і *dist*, а отже, і представлення функцій *push* і *pop*. ■

Приклади. Нехай $X = \{x, y, z, u, v, w\}$ і $p = xyzuv$. Тоді маємо

1) $\mathrm{add}(p, 3, x) = \mathrm{hl}(xyzuv, 3)\, x\mathrm{tr}\, (xyzuv, 3) = head(xyzuv)\mathrm{hl}(\mathrm{tail}(xyzuv), 2)x\mathrm{tr}(yzuv, 2) = x\mathrm{head}(yzuv)\mathrm{hl}(zuv, 1)x\mathrm{tr}(zuv, 1) = xyzxuv$.

2) $\mathrm{sub}(p, 3) = \mathrm{hl}(xyzuv, 2)\ \mathrm{tr}(xyzuv, 3)) = \mathrm{head}(xyzuv)\ \mathrm{hl}(\mathrm{tail}(xyzuv),1)\ \mathrm{tr}(yzuv, 2) = xyuv$.

3) $\mathrm{hl}(p, 4) = \mathrm{head}(xyzuv)\mathrm{hl}(yzuv, 3)) = x\mathrm{head}(yzuv)\mathrm{hl}(zuv, 2) = xyhead\ (zuv)\mathrm{hl}(uv, 1) = xyzu$.

4) $\mathrm{tr}(p, 3) = \mathrm{tail}(\mathrm{tail}(\mathrm{tail}(xyzuv))) = \mathrm{tail}(\mathrm{tail}(yzuv)) = \mathrm{tail}(zuv) = uv$.

5) $\mathrm{dist}(p, 2) = (\mathrm{hl}(p, 2), \mathrm{tr}(p, 2)) = (\mathrm{head}(xyzuv)\mathrm{head}(\mathrm{tail}(xyzuv)), (\mathrm{tail}(\mathrm{tail}(xyzuv))) = (x\mathrm{head}(yzuv), (\mathrm{tail}\ (yzuv))) = (xy, zuv)$.

6) $\mathrm{push}(p, x) = \mathrm{add}(p, 1(p), x) = \mathrm{add}(xyzuv, 5, x) = xyzuvx$.

7) $\mathrm{pop}(p) = \mathrm{sub}(p, 1(p)) = \mathrm{sub}(xyzuv, 5) = xyzu$. ♠

Слід зазначити, що функції *head* і *tail* (*push* і *pop*) дійсно є операціями, у той час як решта функцій (3)—(7) операціями не являються. Це дозволяє ввести таке

Визначення 63. *Універсальна алгебра* $G = (F(X), \Omega = \{conc, head, tail, e\})$ *розширена оператором рекурсії називається* **алгеброю списокових структур (АСС).**

У цій алгебрі легко встановити справедливість таких тотожностей.

- *(а)* $sub(p,l) = tail(p) = tr(p,l)$;
- *(б)* $sub(add(p,i,x),i+l) = p$;
- *(в)* $hl(p,l(p)) = p$;
- *(г)* $tr(p,l(p)) = e$;
- *(д)* $pop(push(p,x)) = p$.

Має місце і ряд інших співвідношень у цій алгебрі.

Деколи разом з множинами функцій (3)—(9) розглядаються і інші корисні функції, такі як

(10) $substr(p, i, j) = x_i \ldots x_{i+j-1}$, тобто це підслово слова p, яке починається з i-го символу і має довжину j;

(11) $conv(p) = x_m x_{m-1} \ldots x_2 x_1$, тобто це перестановка символів, які складають список p у зворотному порядку.

Побудови представлення функцій *substr* і *conv* у вигляді термів алгебри спискових структур пропонуються як вправи.

Тепер, користуючись операціями і функціями (1)—(11), можна вводити й інші функції. Розглянемо деякі з таких функцій.

Нехай задано деякий алфавіт $X = \{x_1, x_2, \ldots, x_n\}$. Множина $F(X)$, як відомо з першого розділу, є повністю упорядкованою відносно лексикографічного порядку, мінімальним елементом якої є пусте слово e. Користуючись цим порядком, визначимо за індукцією такі функції:

а) $l(p) : F(X) \to N$ — функція довжини слова. Індуктивне визначення цієї функції таке:

$$l(p)\begin{cases}0, & \text{коли } p = e\\ 1 + l(tail(p)), & \text{інакше.}\end{cases}$$

б) $subword(p, q) : F(X) \times F(X) \to \{0, 1\}$. Ця функція дорівнює 1, якщо слово q є підсловом слова p, і дорівнює 0 — у протилежному випадку. Індуктивне визначення цієї функції таке: для довільного слова q

$$subword(p,q)\begin{cases}0, & \text{якщо } p = e \text{ і } q \neq e,\\ 1, & \text{якщо } hl(p, l(q)) = q,\\ subword(tail(p), q), & \text{інакше.}\end{cases}$$

Безпосередньо з визначення випливають такі прості властивості: оскільки пусте слово є підсловом довільного слова і довільне непусте слово є підсловом самого себе, то

$$subword(p,e) = 1,\ subword(p,p) = 1,\ subword(p,q) = 0,$$
$$\text{якщо } l(p) < l(q).$$

в) $substit(p, q, r) : F(X) \times F(X) \times F(X) \to F(X)$. Результатом цієї операції є підстановка слова r замість першого входження слова q в слово p. Визначення цієї функції таке: для довільних слів p, q і r

$$substit(p,q,r)\begin{cases}e, & \text{якщо } p = e,\\ r \cdot tr(p, l(q)), & \text{якщо } hl(p, l(q)) = q,\\ head(p)\,substit(tail(p), q, r), & \text{інакше.}\end{cases}$$

Безпосередньо з визначення випливають такі очевидні властивості:

$$substit(p, e, r) = rp, \qquad substit(p, q, q) = p.$$

Приклади. Знайти

а) $subword(abbcda, cd)$

б) $subword(abbc, ax)$

в) $substit(abbcda, cd, a)$

г) $substit(abcd, xa, da)$.

Розв'язок.

а) subword(abbcda,cd) = subword(bbcda,cd) = subword(bcda,cd) = = subword(cda,cd) = 1.

б) subword(abbc,ax) = subword(bbc,ax) = subword(bc,ax) = = subword(c,ax) = subword(e,ax) = 0.

в) substit(abbcda,cd,a) = asubstit(bbcda,cd,a) = absubstit(bcda,cd,a) = = abbsubstit(cda,cd,a) = abbaa.

г) substit(abcd,xa,da) = asubstit(bcd,xa,da) = absubstit(cd,xa,da) = = abcsubstit(d,xa,da) = abcdsubstit(e,xa,da) = abcd. ♠

3.8.2. Алгебраїчна система спискових структур

Розширимо АСС предикатом рівності – =. Розширену таким чином АСС будемо називати **алгебраїчною системою спискових структур (АССС)**. Для цієї алгебраїчної системи має місце

Теорема 76. *АССС є алгоритмічно повною системою, тобто системою, в якій можна обчислити довільну частково рекурсивну функцію.*

Доведення. Згідно з теоремами 73, 74 для доведення необхідно показати, що довільний нормальний алгорифм Маркова можна представити в цій системі.

Нехай Ф — деякий нормальний алгорифм Маркова, заданий системою формул підстановки R в алфавіті X:

$$\begin{cases} p_1 & \to & [.]q_1 \\ p_2 & \to & [.]q_2 \\ \dots & \dots & \dots \\ p_m & \to & [.]q_m \end{cases} \quad \begin{cases} p_1 & \to & q_1' \\ p_2 & \to & q_2' \\ \dots & \dots & \dots \\ p_m & \to & q_m' \end{cases}$$

де $q_i' = .q_i$, якщо формула підстановки заключна і $q_i' = q_i$, якщо формула підстановки проста. Далі, нехай $p \in F(X)$ і $G = (F(X'),$

{*conc*, *head*, *tail*}, {=}) — АССС над алфавітом $X' = X \cup \{.\}$. Тоді, використовуючи операції і предикати АССС, а також функції *subword*, *substit* і функцію довжини, можемо записати для довільного $p \in F(X)$:

$\Phi(p)$ = якщо *subword*$(p, p_1) = 1$ то

якщо *head*(q_1') = «.» то *substit*$(p, p_1,$ *tail*$(q_1'))$

інакше Φ(*substit*(p, p_1, q_1')

інакше якщо *subword*$(p, p_2) = 1$ то

якщо *head*(q_2') = «.» то *substit*$(p, p_2,$ *tail*$(q_2'))$

інакше Φ(*substit*(p, p_2, q_2')

інакше.....

.....

інакше якщо *subword*$(p, p_m) = 1$ то

якщо *head*(q_m') = «.» то *substit*$(p, p_m,$ *tail*$(q_m'))$

інакше Φ(*substit*(p, p_m, q_m')

інакше p.

Покажемо індукцією за числом n застосованих підстановок до слова p, що функція $f(p)$, яка отримана за допомогою системи R, і функція $f'(p)$, яка отримана за допомогою системи $\Phi(p)$, збігаються.

При $n = 0$ маємо $f(p) = p$, оскільки жодна із підстановок системи R не застосовна до слова p. Із системи $\Phi(p)$ випливає, що $f'(p) = p$, оскільки всі умови вигляду *subword*$(p, p_i) = 0$. Отже, у цьому випадку $f(p) = f'(p)$.

Припустимо, що рівність виконується для всіх $m < n$ і нехай n-ою формулою підстановки є формула $p_i \to q_i'$. Можливі такі випадки: формула $p_i \to q_i'$ — заключна і формула $p_i \to q_i'$ не заключна.

Нехай у першому випадку $m = n - 1$ і після виконання m-ї підстановки одержано слово p'. За припущенням індукції $f(p') = f'(p')$. З того, що $p_i \to q_i'$ застосовна до p' випливає, що жодна з підстановок, які їй передують у системі R, не застосовні до p', або що те саме, що всі умови вигляду *subword*$(p', p_j) = 0$, де $j < i$, а умова *subword*$(p', p_i) = 1$ і *head*(q_i') = «.». що $f(p')$ = *substit*$(p', p_i,$ *tail*$(q_i')) =$ *substit*$(p', p_i, q_i) = f'(p')$.

Другий випадок аналогічний першому, з тією лише різницею, що в першому випадку обчислення зупиняються, а в другому — продовжуються. ■

Для ілюстрації роботи системи $\Phi(p)$ розглянемо деякі приклади.

Приклади. а) $X = \{a, b\}$, R має вигляд

$$\begin{cases} a & \to & .e & (= q_1') \\ b & \to & b & (= q_2') \end{cases}$$

Підставивши дані формули в систему $\Phi(p)$, отримуємо

$\Phi(p)$ = якщо *subword*$(p, a) = 1$ то *substit*(p, a, e)
інакше якщо *subword*$(p, b) = 1$ то $\Phi(p)$ інакше p.

З отриманого виразу випливає, що $\Phi(p)$ не припиняє обчислень, коли непусте слово p не має входжень літери a, тобто результат застосування $\Phi(p)$ не буде визначений.

б) $X = \{x, x^{-1}, y, y^{-1}\}$, R має вигляд

$$\begin{cases} xx^{-1} & \to & e \\ x^{-1}x & \to & e \\ yy^{-1} & \to & e \\ y^{-1}y & \to & e. \end{cases}$$

Тоді

Φ(p) = якщо *subword*$(p,\ xx^{-1}) = 1$ то Φ(*substit*$(p,\ xx^{-1}, e)$
інакше якщо *subword*$(p,\ x^{-1}x) = 1$ то Φ(*substit*$(p,\ x^{-1}x, e)$
інакше якщо *subword*$(p,\ yy^{-1}) = 1$ то Φ(*substit*$(p,\ yy^{-1}, e)$
інакше якщо *subword*$(p,\ y^{-1}y) = 1$ то Φ(*substit*$(p,\ y^{-1}y, e)$
інакше p.

в) $X = \{1, *\}$, $Y = \{1, *, \alpha\}$, а R має вигляд

$$\begin{cases} * & \to & * \\ \alpha 1\,1 & \to & \alpha 1 \\ \alpha 1 & \to & .1 \\ e & \to & \alpha. \end{cases}$$

Тоді маємо

Φ(p) = якщо *subword*$(p, *) = 1$ то $\Phi(p)$
інакше якщо *subword*$(p, \alpha 1\,1) = 1$ то Φ(*substit*$(p, \alpha 1\,1, \alpha 1)$
інакше якщо *subword*$(p, \alpha 1) = 1$ то *substit*$(p, \alpha 1, 1)$
інакше $\Phi(\alpha p)$. ♠

3.8.3. Застосування нормальних алгорифмів

Розглянемо приклад застосування теорії алгорифмів Маркова для доведення алгоритмічної повноти мови програмування ЛІСП.

1. Короткий огляд функціональної мови програмування ЛІСП

Мова ЛІСП являє собою мову функціонального програмування [41], яка використовує *символи* і побудовані з них *символьні структури*.

Символ — це ім'я, що складається із літер, цифр і спеціальних знаків, які означають який-небудь предмет, об'єкт, річ або дію реального світу. У ЛІСП символи можуть означати числа, інші символи або більш складні структури, програми (функції) та інші об'єкти ЛІСП. Наприклад, символ «+» може означати функцію додавання, ізотоп вуглецю — «вуглець-14» і т.д.

Приклад символів. *x*, Символ, *defun*, *step* — 1987. ♠

Символи в ЛІСП можуть складатися з літер, цифр і деяких інших знаків, таких як

$$+, -, *, /, @, \$, \%, \&, \_, \backslash, <, >, \sim.$$

Знаки оклику і питання, як і тильда (~) і квадратні дужки, теж можуть входити до складу символів, але ці знаки служать для спеціального використання.

Числа є константами, а **T** і **NIL** — логічними значеннями, які відповідають істині і хибності. Символом NIL позначають також пустий список. Основою мови програмування ЛІСП є так званий COMMON Lisp.

Атомами в COMMON Lisp є символи і числа, які складають алфавіт *A* мови ЛІСП. **Списком** називається упорядкована послідовність, елементами якої є атоми або списки (підсписки-підслово). Списки заключаються в круглі дужки, наприклад: (*b*(*cd*)*e*). Отже, список у мові ЛІСП — це ієрархічна структура даних.

Приклад. (+23) — список із трьох елементів,

(((((перший) 2) третій) 4) 5) — список із двох елементів,

(Добрий день сказав бородатий чоловік) — список із п'яти елементів,

(Кіт–37 (кличка Фіма) (колір ?) (хвіст *NIL*)) — список із чотирьох елементів,

NIL — пустий список. ♠

2. Поняття функції в ЛІСП. Базові функції ЛІСПа

Функцією в ЛІСП називається відображення f: $F(A) \to F(A)$, яке записують у вигляді $y = f(x)$, де $x \in F(A)$, $y \in F(A)$. Базовими функціями у мові програмування ЛІСП є такі функції: **CAR, CDR, CONS, ATOM** та **EQ.**

Функції **CONS, EQ** бінарні, а решта функції унарні.

Функції **ATOM, EQ** являють собою предикати.

Семантика цих функцій така.

Функція **CAR** — це відома нам функція *head*, тобто $CAR(p) = head(p) = x_1$, де $p = x_1 \ldots x_k$.

Функція **CDR** — це відома нам функція *tail*, тобто $CDR(p) = tail(p) = x_2 \ldots x_k$.

Функція **CONS** — це конкатенація двох списків, один з яких (перший аргумент) одноелементний список: $CONS(a, p) = ap$.

Предикат **ATOM** перевіряє, є чи ні даний елемент атомом. Наприклад, $() = 1$, $(p) = 0$, якщо $l(p) > 1$.

Предикат **EQ** порівнює два символи і якщо вони однакові то дорівнює 1, інакше дорівнює 0. Насправді у ЛІСП використовується більш загальний предикат, який працює так само як і EQ, але додатково дає можливість порівнювати числа одного й того самого типу (і елементи слів).

Існують і інші функції і предикати в ЛІСП, які відображуються через базові функції, тому вони не розглядаються.

Із теореми про алгоритмічну повноту алгебраїчної системи списковых структур випливає

Теорема 77 (*Теорема про алгоритмічну повноту ЛІСП*). *Мова функціонального програмування ЛІСП є алгоритмічно повною мовою.*

Доведення очевидне, оскільки $CAR = head$, $CDR = tail$, $CONS = conc$, $EQ =$ "=" і $NIL = e$

3.9. Контрольні питання, задачі і вправи

Контрольні питання

1. Дайте визначення алгоритмічно повної мови програмування.
2. Які Ви знаєте функціональні мови програмування?
3. Чи є алгоритмічно повною мова програмування ЛІСП? (Відповідь обґрунтуйте).

Задачі і вправи

1. Представте предикат рівності АССС у вигляді програми мовою ЛІСП.
2. Напишіть програму мовою ЛІСП для перевірки слова на властивість бути паліндромом.

3. Дайте повне доведення теореми про алгоритмічну повноту мови ЛІСП.

4. Подайте функції *sub*, *add*, *subword*, *substit* у вигляді програм мовою ЛІСП.

5. Чи буде алгоритмічно повною система списокових структур, яка включає лише такі базові операції і предикати: *head*, *tail*, *add* і = ?

6. Побудуйте програми обчислення найпростіших функцій $o(x)$, $s(x)$ і $I_m^n(x_1,..., x_n)$ в АССС і мовою ЛІСП.

7. Навести приклад частково рекурсивної повністю визначеної функції, застосування оператора мінімізації до якої дає частково визначену функцію.

8. Довести, що нижченаведені функції можна отримати із функцій $s(x) = x + 1$ і $q(x) = x - [\sqrt{x}]^2$ за допомогою операторів суперпозиції, додавання двох функцій і ітерації ($[\sqrt{x}]$ — найбільше ціле число, яке не більше $\sqrt{x}$, а функція f, яка одержується із функції g за допомогою ітерації визначається так: $f(0) = 0, f(x + 1) = g(f(x))$):

a) $o(x)$; b) $I_m^n(x_1,..., x_n)$, c) $sg(x)$; d) $\overline{sg}(x)$;

e) $x \cdot y$; f) $x \div y$; g) $[\sqrt{x}]$; h) x^2.

9. Побудувати машину Тьюрінга, яка виконує

a) перенос нуля $0q_0 0\underbrace{1...1}_{n}0 \to 0q\underbrace{1...1}_{n}00$;

b) правий зсув $0q_0\underbrace{1...1}_{n}0 \to 0q\underbrace{1...1}_{n}$;

c) лівий зсув $0q_0\underbrace{1...1}_{n} \to 1q\underbrace{1...1}_{n-1}0$;

d) транспозиція $0\underbrace{1...1}_{n}0q_0\underbrace{1...1}_{m}0 \to \underbrace{1...1}_{m}0q\underbrace{1...1}_{n}0$;

e) подвоєння $0q_0\underbrace{1...1}_{n}0 \to 0q\underbrace{1...1}_{n}0\underbrace{1...1}_{n}0$;

f) циклічний зсув

$$0q_0\underbrace{1...1}_{m}0\underbrace{1...1}_{n}0...0\underbrace{1...1}_{k}0 \to 0q\underbrace{1...1}_{n}0...0\underbrace{1...1}_{k}0\underbrace{1...1}_{m}0;$$

10. Побудувати машину Тьюрінга, яка обчислює $I_m^n\{x_1,..., x_n)$.

11. Довести теорему 61.

4 ЕЛЕМЕНТИ ТЕОРІЇ СКЛАДНОСТІ ОБЧИСЛЕНЬ

З наведених вище прикладів алгоритмів випливає, що значення обчислювальної функції можуть бути обчислені за допомогою різних алгоритмів, які збудовані з найпростіших функцій, чи за допомогою різних машин Поста, чи за допомогою різних машин Тьюрінга. Загалом для кожної з цих алгоритмічних систем існує нескінченне число «програм», які обчислюють значення даної обчислювальної функції при заданих значеннях аргументів. Природно після уточнення поняття «алгоритм» зробити наступний крок — навчитися порівнювати різні алгоритми між собою за величиною трудозатрат. Основою для такого порівняння може служити довільна із розглянутих алгоритмічних систем. Вибір припав на машини Тьюрінга і в цьому розділі розглядаються основні поняття теорії складності обчислень за Тьюрінгом, методи аналізу алгоритмів і класифікація складності обчислень.

4.1. Основні поняття

Одним з основних завдань теорії складності обчислень є класифікація проблем відповідно до складності їх обчислень за допомогою комп'ютера. Це означає, що така теорія для заданої проблеми повинна відповідати на питання: якими мають бути потужність комп'ютера і його ресурси для того, щоб на цьому комп'ютері можна було розв'язати дану проблему, і який із двох алгоритмів, що розв'язують одну й ту саму проблему, найбільше підходить? На жаль, вичерпної відповіді на це питання в даний момент не існує, але існує збудована певна класифікація проблем відповідно до їх складності. Далі будуть розглянуті найбільш популярні класи складності, які часто зустрічаються в теорії і на практиці.

4.1.1. Означення проблеми

Спочатку зафіксуємо алфавіт $Al = \{0, 1\}$ і напівгрупу всіх слів скінченної довжини в цьому алфавіті $\{0, 1\}^*$.

Визначення 64. **Проблемою** *називається множина X упорядкованих пар вигляду (I, A) слів у алфавіті* $\{0, 1\}$, *де I називається* **прикладом,** *а A називається* **відповіддю** *для цього прикладу і кожне слово* $I \in \{0,1\}^*$ *є першим компонентом хоча б для однієї такої пари.*

З цього означення випливає, що проблема є повністю визначеним бінарним відношенням на множині всіх слів у алфавіті *Al*. Крім того, це означення є досить абстрактним, але його формулювання в такому вигляді необхідне для того, щоб у подальшому класифікацію проблем за їх класами складності і всіх пов'язаних з нею понять можна було обговорювати в термінах машинних моделей. Перед тим як перейти до означення цих понять, розглянемо приклад проблеми.

Приклад проблеми ізоморфізму графів

ПРОБЛЕМА: Дано два неорієнтовані графи $G = (V, E)$ і $G' = (V', E')$, де V і V' є скінченними множинами вершин, а E і E' — скінченними множинами ребер.

ПИТАННЯ: Чи є ці два графи ізоморфними?

ВІДПОВІДЬ: *«yes»*, коли існує взаємно однозначне відображення $f: V \to V'$ таке, що із включення $(u, v) \in E$ випливає включення $(f(u), f(v)) \in E'$. У протилежному випадку — відповідь *«no»*.

З означення проблеми ізоморфізму графів випливає, що ця проблема матиме сенс тільки тоді, коли розв'язане питання представлення графів у вигляді слів у алфавіті $\{0, 1\}$. Іншими словами, питання формулюється таким чином: чи представляють два словникові подання деяких графів один і той самий граф, чи ні? Для формулювання проблеми введемо символи a_Y і a_N для надання відповіді *«yes»* і *«no»* відповідно. Тоді відношення на словах буде таким:

$\{(p, a_Y)$: слово p являє собою два записи одного й того самого графа $G\} \cup$
$\cup \{(p, a_N)$: слово p не являє собою два записи одного й того самого графа $G\}$.

Аналогічно може бути сформульовано і довільну іншу проблему і в цьому є необхідність, оскільки ми хочемо розв'язати її за допомогою комп'ютера. ♠

Визначення 65. *Функцією будемо називати відношення на множині слів у алфавіті* $\{0, 1\}$, *в якому кожне слово* $p \in \{0, 1\}^*$ *є першим компонентом у точності однієї пари* (p, A).

Розв'язуваною (decision) проблемою *називається функція, в якої відповідями є тільки два значення «yes» або «no». І якщо для прикладу I отримується відповідь «yes», то такий приклад називається* **позитивним прикладом** *проблеми* (I, A).

Обчислювальною (counting) проблемою *називається функція, в якої всі відповіді є невід'ємними цілими числами.*

Розв'язувані проблеми є важливим окремим випадком обчислювальних проблем, оскільки більшість класів складності обмежуються тільки такими проблемами або, більш точно, мовами, які отримуються з них. Нагадаємо поняття мови.

Визначення 66. **Мовою** L *в алфавіті* $\{0, 1\}$ *називається довільна підмножина множини всіх слів скінченної довжини в цьому алфавіті, тобто* $L \subseteq \{0, 1\}^*$.
Якщо L — мова, то розв'язуваною проблемою, що відповідає мові L, є множина

$$\{(p, «yes») : p \in L\} \cup \{(p, «no») : p \notin L\}.$$

Мовою $L(R)$, *що відповідає розв'язуваній проблемі R, називається множина слів*

$$L(R) = \{p \in \{0, 1\}^* : (p, «yes») \in R\}.$$

Зауважимо, що між відповідями *«yes»* і *«no»* існує певна асиметрія, а саме, відповідь *«no»* не може з'явитись, якщо з'явилася відповідь *«yes»*, і навпаки. Заміна відповіді *«yes»* на *«no»* приводить до поняття мови-доповнення, означення якого наводиться нижче.

Визначення 67. *Якщо L — деяка мова в алфавіті* $\{0, 1\}$, *то її мовою-доповненням або просто доповненням називається мова* $Co(L) = \{0,1\}^* \setminus L$.

Ця асиметрія в означенні мови $L(R)$ має фундаментальне теоретичне значення. Справа в тому, що для багатьох проблем R мови $L(R)$ і $Co(L(R))$ не завжди належать до одного й того самого класу складності. У зв'язку з цим буде розв'язуватися питання: чи є заданий клас замкнутим «відносно доповнення», тобто чи вірно, що для деякого класу складності C, з того що $L \in C$ випливає, що і $Co(L) \in C$ для всіх мов L?

4.1.2. Методи розв'язання проблем

У теорії складності обчислень стверджують, що деяка проблема розв'язувана тільки тоді, коли існує загальний метод, який застосовний до довільного прикладу цієї проблеми, за наявності достатнього часу, пам'яті та інших необхідних ресурсів. На практиці такий метод часто буває корисним тільки для розв'язання досить невеликої кількості прикладів через фізичну обмеженість доступних ресурсів. Але з огляду на те, що метод розв'язання проблеми є загальним, він здатний автоматично розв'язувати приклади великих розмірів у межах доступних ресурсів. Тому головним питанням, яке виникає при розв'язанні деякої проблеми, є таке: які ресурси необхідні для того, щоб можна було розв'язати конкретний приклад проблеми або проблему в цілому?

Визначення 68. **Розміром прикладу** ***I****, який позначається через* $l(I)$*, називається його довжина, тобто кількість символів у слові* I *(нагадаємо, що відповідно до формального означення, проблеми є словами в алфавіті* $\{0, 1\}$*).*

Визначення 69. *Якщо* M *є методом розв'язання проблеми* X *і* R *є необхідним ресурсом для методу* M*, то* $R_M : N^+ \to N^+$ *є функцією, що визначає величину* $R_M(n)$ *як максимум по всіх словах* p *довжини* n *серед усіх ресурсів* R*, необхідних для застосування* M *до прикладу* p*.*

Зауважимо, що $R_M(n)$ є величиною-ресурсом, який потрібний у найгіршому випадку. В окремих випадках приклад p довжини $l(p) = n$ і метод M можуть вимагати набагато менших ресурсів, ніж $R_M(n)$. Але величина $R_M(n)$ гарантує те, що метод M буде застосовним до довільного прикладу p довжини n.

Точно визначити поняття «ресурсні вимоги» тієї чи іншої проблеми досить важко. Через те це поняття вводять за допомогою деяких непрямих понять і означень, основні з яких це поняття верхньої і нижньої границь складності. Нехай X — деяка проблема, R — необхідний ресурс для її розв'язання і C — клас методів розв'язання.

Визначення 70. *Вимоги проблеми* X *до ресурсу* R *відносно методів із класу* C *мають* **верхню границю** $T(n)$ *тоді і тільки тоді, коли існує метод* $M \in C$ *для розв'язання проблеми* X*, такий, що* $R_M(n) = O(T(n))$*. Вимоги проблеми* X *до ресурса* R *відносно методів із класу* C *мають* **нижню границю** $T(n)$ *тоді і тільки тоді, коли для всіх методів* $M \in C$ *для розв'язання проблеми* X *істинна рівність* $R_M(n) = O(T(n))$*.*

Як випливає з цих означень, поняття вимог до ресурсу визначаються в термінах «клас методів» і «метод із класу». Ці поняття в подальшому будуть уточнятися, а зараз розглянемо машинні моделі, у термінах яких будується ієрархія класів складності.

4.2. Машинні моделі

Класи методів і класи складності найбільш природно описуються за допомогою машинних моделей обчислень. Стандартним варіантом таких моделей є машини Тьюрінга, такі як детерміновані, недетерміновані, багатострічкові та інші їх варіації.

4.2.1. Детерміновані машини Тьюрінга

Неформально детермінована машина Тьюрінга (ДМТ) являє собою пристрій, який має чуттєву **головку** G, що здатна пересуватися вздовж нескінченної в один (правий) бік стрічки, яка розбита на **клітинки.** Правила руху головки вздовж стрічки визначаються її **програмою** P, яка в теорії ДМТ називається **функцією переходів.** Головка здатна **читати символ із стрічки**, **писати символ на стрічку** і **пересуватися** вліво чи вправо на одну клітинку, або залишатися в тій самій позиції відповідно до програми даної ДМТ.

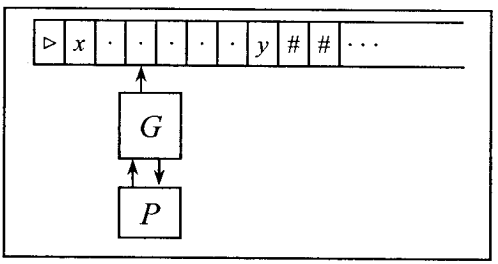


Схема ДМТ

Формальне означення ДМТ є таким.

Визначення 71. *Детермінованою машиною Тьюрінга називається упорядкована четвірка* $M = (K, X, \delta, s_0)$, *де* K *— скінченна множина, елементи якої називаються* **станами ДМТ**, X *— скінченна множина попарно різних елементів, що називається* **алфавітом ДМТ** (*причому* $K \cap X = \varnothing$, *а алфавіт* X *включає символи* # (*пустий*

символ), $\triangleright$ (*початковий символ*) *і пусте слово* e), $\delta : K \times X \to K' \times X \times \{l, r, t\}$ — **функція переходів** *ДМТ*, *де* $K' = K \cup \{h, «yes», «no»\}$ *і* $s_0 \in K$ — **початковий стан ДМТ.**

Символи h, *«yes»*, *«no»* означають відповідно **заключний стан (або фінальний), стан, що сприймає, і стан, що не сприймає**, а символи l, r, t означають напрямок руху головки ДМТ відповідно вліво, вправо і залишатися в тій самій позиції. Нехай X^* означає множину всіх скінченних слів у алфавіті X.

Робота ДМТ відбувається таким чином. Деяке слово $q = \triangleright\, p \in X$ записано на стрічці ДМТ. У початковий момент часу ДМТ знаходиться в початковому стані s_0 і оглядає перший символ слова q — символ $\triangleright$. При цьому слово $p \in X^*$ називається **вхідним словом** або **вхідними даними ДМТ**. Виходячи зі своєї початкової комбінації, ДМТ виконує крок обчислень таким способом. Для кожної комбінації поточного стану $k \in K$ і символу $x \in X$, функція δ визначає трійку (k', y, d), де k' — черговий стан, $y \in X$ — символ, який записується на стрічці замість символу x, а $d \in \{l, r, t\}$ визначає напрямок руху головки ДМТ. У випадку, коли робота ДМТ тільки починається, завжди $\delta(s_0, \triangleright) = (k', \triangleright, r)$, тобто символ $\triangleright$ завжди призводить до руху головки ДМТ вправо і на його місце не можна нічого записувати. Ця обставина дозволяє розглядати ДМТ зі стрічкою, необмеженою в один (правий) бік, оскільки вихід за початковий символ $\triangleright$ у такій ситуації неможливий.

Не дивлячись на те, що головка не може вийти за лівий кінець вхідного слова p, вона може вийти за його правий кінець. У цьому випадку говорять, що головка читає пустий символ #, на місце якого може бути записаний якийсь символ з алфавіту X. Таким чином, послідовність символів на стрічці може зростати, що необхідно для того, щоб виконувати довільні обчислення за допомогою ДМТ.

Зупинка ДМТ настає тоді, коли вона в процесі обчислень досягає одного з трьох станів h, *«yes»*, *«no»*. При цьому, якщо ДМТ зупинилась

— у стані *«yes»*, то говорять, що ДМТ **сприймає** або **розпізнає вхідне слово** $p \in X$;

— у стані *«no»*, то говорять, що ДМТ **не сприймає** або **не розпізнає вхідне слово** $p \in X$;

— у стані h, то результатом роботи ДМТ вважається слово, що записане на стрічці ДМТ у момент її зупинки.

Оскільки обчислення у разі зупинки ДМТ відбуваються в скінченному часі, то на стрічці буде записано слово скінченної довжини, яке починається символом $\triangleright$ і закінчується символом, який передує першому пустому символу, якщо дивитися на символи слова зліва

направо (це слово може бути, зокрема, і пустим). Якщо слово q є результатом роботи ДМТ на вхідному слові p, то пишемо $M(p) = q$.

Може трапитися й таке, що ДМТ ніколи не досягне якого-небудь із станів h, «yes», «no» і тому її робота буде продовжуватися нескінченно. У цьому випадку вважається, що результат роботи ДМТ на вхідному слові p **невизначений** або, що ДМТ **незастосовна до вхідного слова** p.

Приклад 4.2.1. Розглянемо ДМТ $M = (K, X, \delta, s_0)$, де $K = \{s_0, s, s_1, s_2\}$, $X = \{0, 1, \#, \triangleright\}$, а функція переходів даної ДМТ δ задається такою таблицею:

$k \in K$	$x \in X$	$\delta(k, x)$
s_0	0	$(s_0, 0, r)$
s_0	1	$(s_0, 1, r)$
s_0	#	$(s, \#, l)$
s_0	$\triangleright$	$(s_0, \triangleright, r)$
s	0	$(s_1, \#, r)$
s	1	$(s_2, \#, r)$
s	#	$(s, \#, l)$
s	$\triangleright$	$(h, \triangleright, r)$
s_1	0	$(s_0, 0, l)$
s_1	1	$(s_0, 0, l)$
s_1	#	$(s_0, 0, l)$
s_1	$\triangleright$	$(h, \triangleright, r)$
s_2	0	$(s_0, 1, l)$
s_2	1	$(s_0, 1, l)$
s_2	#	$(s, 1, l)$
s_2	$\triangleright$	$(h, \triangleright, r)$

Таблиця функції переходів

0)	s_0	$\underline{\triangleright}010$
1)	s_0	$\triangleright\underline{0}10$
2)	s_0	$\triangleright0\underline{1}0$
3)	s_0	$\triangleright01\underline{0}$
4)	s_0	$\triangleright010\underline{\#}$
5)	s	$\triangleright01\underline{0}\#$
6)	s_1	$\triangleright01\underline{\#}\#$
7)	s_0	$\triangleright01\underline{\#}0$
8)	s	$\triangleright0\underline{1}\#0$
9)	s_2	$\triangleright0\underline{\#}\#0$
10)	s_0	$\triangleright0\underline{\#}10$
11)	s	$\triangleright\underline{0}\#10$
12)	s_1	$\triangleright\underline{\#}\#10$
13)	s_0	$\triangleright\underline{\#}010$
14)	s	$\underline{\triangleright}\#010$
15)	h	$\triangleright\underline{\#}010$

Послідовність обчислень ДМТ

Якщо застосувати цю ДМТ до вхідного слова $p = 010$, то отримуємо послідовність обчислень, яка показана праворуч від таблиці функції переходів. Як видно з результатів роботи цієї ДМТ, вона вставляє пустий символ # між початковим символом $\triangleright$ і вхідним словом p. Аналіз програми даної ДМТ показує, що якщо вхідне слово p не має входжень пустого символу, то ДМТ завжди зупиняється. Якщо ж слово p має входження хоча б одного пустого символу, то в її програмі стає досяжним стан (s, #, t) і оскільки $\delta(s, \#) = (s, \#, t)$, то ДМТ ніколи не зупиниться і буде працювати нескінченно довго. Це означає, що дана ДМТ застосовна до довільного вхідного слова, яке не має входжень пустого символу #, і незастосовна до вхідних слів, які

мають входження пустого символу. Слід зауважити, що значення $\delta(s, \#)$ можна було б визначити так, щоб ДМТ зупинялась і генерувала, наприклад, відповідь «no». ♠

Зауваження. Як випливає з даного означення, поняття ДМТ відрізняється від загальноприйнятого (див. параграф 3.4). В загальноприйнятому означенні ДМТ стрічка цієї ДМТ нескінченна в обидва боки. Покажемо, що наведене поняття ДМТ з нескінченною в один бік стрічкою не звужує обчислювальної потужності ДМТ з нескінченною в обидва боки стрічкою. Має місце таке твердження.

Теорема 78. *Довільна ДМТ M з нескінченною в обидва боки стрічкою моделюється еквівалентною їй ДМТ M' з нескінченною в один (правий) бік стрічкою, і навпаки.*

Доведення. Те, що ДМТ M' з нескінченною в один бік стрічкою моделюється відповідною ДМТ M з нескінченною в обидва боки стрічкою, очевидно, оскільки ДМТ M є узагальненням ДМТ M'. Доведемо обернене твердження.

Нехай $M = (K, X, \delta, s_0)$ — ДМТ з нескінченною в обидва боки стрічкою (див. рис. ДМТ)

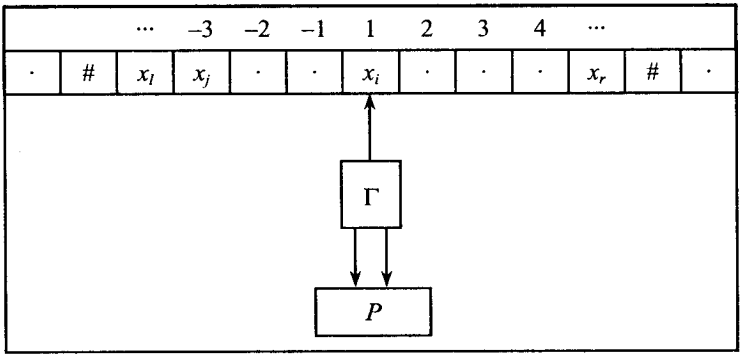


Схема ДМТ з нескінченною в обидва боки стрічкою

Побудуємо ДМТ $M' = (K', X \times X, \delta, s_0)$ з нескінченною в один бік стрічкою, яка поділена на дві доріжки — *верхню* і *нижню*. Верхня доріжка відповідає лівій половині стрічки ДМТ M, починаючи з номера –1, а нижня — правій половині стрічки.

У кожній клітинці двохдоріжкової стрічки записана пара символів $(x_{-i}, x_i) \in X \times X$, де X — алфавіт ДМТ M.

Опишемо спочатку неформально роботу ДМТ M', яка моделює роботу ДМТ M. У разі описаного представлення стрічки ДМТ M на стрічці ДМТ M', головка ДМТ M' переміщується в точній відповідності з програмою ДМТ M вздовж своєї нижньої доріжки без якихнебудь змін символів на верхній доріжці. Так відбувається доти, поки головка ДМТ M' знаходиться над клітинками x_i, у яких $i \geq 1$. При переході головки ДМТ M' вліво від клітинки 1 ДМТ M' читає символ $\triangleright$ — кінець стрічки, після чого її головка переміщується по верхній доріжці в бік, протилежний переміщенням головки ДМТ M, по тих клітинках x_{-i}, для яких $i \geq 1$. При цьому вміст клітинок нижньої доріжки не змінюється.

Формальне означення ДМТ M' має такий вигляд. Кожному стану $q_i \in K$ відповідають два стани q_i^U і q_i^L, верхні індекси яких свідчать про те, з якою доріжкою працює в даний момент ДМТ M'. Нехай для ДМТ M має місце така рівність:

$$\delta(q_i, a) = (q_j, b, d),$$

де $q_i, q_j \in K$, $a, b \in X$, $d \in \{l, t, r\}$. Тоді для ДМТ M' при всіх $v \in X$ припускаємо

$$\delta(q_i^L, (a, v)) = (q_i^L, (b, v), d^+)$$

і

$$\delta(q_i^U, (a, v)) = (q_i^U, (v, b), d^+),$$

де $d^+ = t$, якщо $d = t$, $d^+ = l$, якщо $d = l$ і $d^+ = r$, якщо $d = r$. Залишається додати ще один перехід для початкового символу $\triangleright$:

$$\delta(q_i^L, \triangleright) = (q_i^U, \triangleright, r)$$

і

$$\delta(q_i^U, \triangleright) = (q_i^L, \triangleright, r).$$

При цьому переміщення головки вліво із самої лівої клітинки верхньої чи нижньої доріжки призведе до переходу на іншу доріжку, тобто до зміни q_i^L на q_i^U, і навпаки.

Доведення того, що для кожного вхідного слова $p \in F(X)$ має місце рівність $M(p) = M'(p)$, пропонується як вправа. ■

Завдяки цій теоремі в подальшому будемо користуватися лише поняттям ДМТ з нескінченною в один (правий) бік стрічкою, оскільки обчислювальна потужність ДМТ при цьому не змінюється.

Операції, що виконуються ДМТ у процесі обчислень, можна визначити формально за допомогою поняття **конфігурації ДМТ**.

Визначення 72. *Конфігурацією ДМТ називається трійка* (s, p, q), *де* $s \in K$, *а слова* p *і* q *із* X^* *такі, що* p *є словом, яке записане на стрічці ДМТ зліва від клітинки, яка оглядається головкою, включаючи і символ, що записаний у цій клітинці, а* q *— слово, записане на стрічці праворуч від клітинки, що оглядається головкою ДМТ.*

Конфігураціями ДМТ у наведеному вище прикладі 4.2.1 служать такі трійки: початкова конфігурація $(s_0, \triangleright, 010)$, після якої ДМТ послідовно переходить до таких конфігурацій:

$$(s_0, \triangleright\, 0, 10), (s_0, \triangleright\, 01, 0), (s_0, \triangleright\, 010, e),$$
$$(s_0, \triangleright\, 010\#, e), (s, \triangleright\, 010, \#), (s_1, \triangleright\, 01\#\#, e)$$

і т. д.

Визначення 73. *Нехай* M *деяка ДМТ. Стверджують, що ДМТ* M *з конфігурації* (s, p, q) *безпосередньо досягає конфігурації* (s', p', q') *(позначення* $(s, p, q) \xrightarrow{M} (s', p', q')$*), якщо виконуються такі умови:*
а) $\delta(s, x) = (s', y, d)$, *якщо* x *— останній символ слова* p;
b) якщо $d = l$, *то* p' *є словом, яке одержане із слова* p *шляхом викреслювання символа* x, *а* $q' = yq$;
с) якщо $d = r$, *то* $p' = py$, *а* q' *є словом, яке одержане із слова* q *шляхом викреслювання його першого символу;*
в) якщо $d = t$, *то* p' *є словом, у якого останній символ* x *замінений на символ* y, *а* $q' = q$.

Нехай M^* означає транзитивне замикання відношення безпосередньої досяжності для конфігурацій ДМТ M (позначення $(s, p, q) \xrightarrow{M^*} (s', p', q')$). Відношення M^* називається просто відношенням досяжності для конфігурацій. Це відношення означає, що існує таке натуральне число $n \in N$, для якого справедливі такі переходи:

$$(s, p, q) \xrightarrow{M^n} (s', p', q'),$$

тобто для всіх $i = 1, 2, \ldots, n$ маємо $s = s_1$, $s_{n+1} = s'$, $p_1 = p$, $q_{n+1} = q'$ і

$$(s_i, p_i, q_i) \xrightarrow{M} (s_{i+1}, p_{i+1}, q_{i+1}).$$

У цьому випадку говорять також, що ДМТ із конфігурації (s, p, q) досягає або переходить до конфігурації (s', p', q') за n кроків.

Приклад 4.2.2. Для ДМТ з попереднього прикладу маємо

$$(s_0, \triangleright, 010) \xrightarrow{M} (s_0, \triangleright\, 0, 10)$$

$$(s_0, \triangleright, 010) \xrightarrow{M^{15}} (h, \triangleright\#, 010),$$

звідки випливає, що

$$(s_0, \triangleright, 010) \xrightarrow{M^*} (h, \triangleright\#, 010),$$

а наведена нижче ДМТ обчислює значення найпростішої примітивно рекурсивної функції $s(n) = n + 1$, де числа n і $n + 1$ записуються в двійковій системі числення. Нехай $M = (K = \{s_0, s\}, X = \{0, 1, \#, \triangleright\})$, а функція δ задається такою таблицею:

$k \in K$	$x \in X$	$\delta(k, x)$
s_0	0	$(s_0, 0, r)$
s_0	1	$(s_0, 1, r)$
s_0	#	$(s, \#, l)$
s_0	$\triangleright$	$(s_0, \triangleright, r)$
s	0	$(h, 1, t)$
s	1	$(s, 0, l)$
s	#	$(s, \triangleright, r)$
Таблиця	функції	переходів

Зауважимо, що дана ДМТ на вхідному слові $p = 111$ видає результат 000, який є неправильним. Це відбувається через те, що настає так зване «переповнення розрядної сітки». Для того, щоб ДМТ правильно обчислювала результат, необхідно щоб спочатку була застосована ДМТ з попереднього прикладу, яка вставляє пустий символ # між початковим символом $\triangleright$ і словом $p = 111$. Після цього дана ДМТ буде правильно обчислювати значення функції $s(n)$.♠

Приклад 4.2.3. Розглянемо ще один приклад ДМТ, яка нічого не обчислює, а тільки сигналізує про закінчення своєї роботи зупинкою в одному із станів «*yes*» або «*no*». Ця ДМТ розпізнає, є чи ні дане слово в алфавіті $X = \{0, 1, \triangleright, \#\}$ паліндромом. Наприклад, для цієї ДМТ маємо

$$M(011110) = \text{«}yes\text{»} \text{ і } M(01101) = \text{«}no\text{»}.$$

ДМТ працює таким чином. У початковому стані s_0 вона шукає в слові, що записане на стрічці ДМТ, перший символ і знайшовши

такий символ, замінює його на символ $\triangleright$ і одночасно запам'ятовує його у своєму стані (наприклад, у стані k_0 символ 0, а в стані k_1 символ 1). Після цього ДМТ шукає перший пустий символ і, знайшовши його, вертається на один крок ліворуч. Якщо останній символ слова збігається з першим, то ДМТ замінює його пустим символом #. Потім ДМТ шукає перший символ $\triangleright$, тобто повертається до початку слова і, знайшовши його, пересувається на один крок вправо. З цього моменту весь процес роботи ДМТ повторюється спочатку. У результаті роботи ДМТ можуть виникнути такі ситуації:

— слово, яке знаходиться між символом $\triangleright$ і пустим символом #, є пустим словом; тоді ДМТ зупиняється в стані «*yes*» — висхідне слово є паліндромом;

— вхідне слово є односимвольним (тобто його кінець і початок збігаються), тоді ДМТ зупиняється в стані «*yes*» — слово є паліндромом;

— якщо в деякий момент останній символ відрізняється від першого в поточному слові, то ДМТ зупиняється в стані «*no*» — висхідне слово не є паліндромом.

Програма цієї ДМТ має такий вигляд:

$k \in K$	$x \in X$	$\delta(k, x)$
s_0	0	$(k_0, \triangleright, r)$
s_0	1	$(k_1, \triangleright, r)$
s_0	$\triangleright$	$(s_0, \triangleright, r)$
s_0	#	(«*yes*», #, *t*)
k_0	0	$(k_0, 0, r)$
k_0	1	$(k_0, 1, r)$
k_0	#	$(k_0', 0, l)$
k_1	0	$(k_1, 0, r)$
k_1	1	$(k_1, 1, r)$
k_1	#	$(k_1', \#, l)$

$k \in K$	$x \in X$	$\delta(k, x)$
k_0'	0	$(k_2, \#, l)$
k_0'	1	(«*no*», 1, *t*)
k_0'	$\triangleright$	(«*yes*», #, *r*)
k_1'	0	(«*no*», 1, *t*)
k_1'	1	$(k_2, \#, l)$
k_1'	#	$(k_0', \#, l)$
k_2	0	$(k_2, 0, l)$
k_2	1	$(k_2, 1, l)$
k_2	#	$(s_0, \triangleright, r)$
Таблиця	функції	переходів

4.2.2. Машини Тьюрінга як алгоритми

Машини Тьюрінга є природними моделями для розв'язання проблем на словах, а саме, **обчислення функцій на словах**, а також **розпізнавання** і **розв'язуваності мов.**

Наведемо формальні означення.

Визначення 74. *Нехай* $L \subseteq (X \setminus \{\#\})^*$ *— деяка мова в алфавіті* $X' = X \setminus \{\#\}$, *а* M *— машина Тьюрінга така, що для довільного слова* $p \in (X \setminus \{\#\})^*$ *отримуємо* $M(p) =$ *«yes», якщо* $p \in L$, *і* $M(p) =$ *«no», якщо* $p \notin L$. *У цьому випадку говорять, що ДМТ* M **розв'язує мову** L.
Мова L *називається* **рекурсивною**, *якщо існує ДМТ, яка розв'язує цю мову.*
Говорять, що ДМТ M *тільки* **розпізнає мову** L, *якщо для довільного слова* $p \in (X \setminus \{\#\})^*$ *маємо* $M(p) =$ *«yes» тоді і тільки тоді, коли* $p \in L$ *і* M *незастосовна до* p, *якщо* $p \notin L$.
Мова L *називається* **рекурсивно перелічимою**, *якщо вона розпізнається деякою ДМТ.*

Найпростіший зв'язок між рекурсивними мовами і рекурсивно перелічимими мовами виражається таким твердженням.

Теорема 79. *Якщо мова* L *рекурсивна, то вона є і рекурсивно перелічимою мовою.*

Доведення. Нехай L є рекурсивною мовою. Тоді існує ДМТ M, яка розв'язує цю мову L. Побудуємо ДМТ M', яка розпізнає мову L, у такий спосіб. M' працює в точності так, як і ДМТ M, за винятком ситуації, коли M має намір зупинитися в стані *«no»*. У цій ситуації M' рухається весь час вправо і ніколи не зупиняється. ■

Машини Тьюрінга застосовуються не тільки для розв'язуваності й розпізнавання мов, але і для обчислення значень словникових функцій.

Визначення 75. *Функція* $f: (X \setminus \{\#\})^* \to X^*$, *де* X *— деякий алфавіт, називається* **словниковою функцією**.
Нехай M *деяка ДМТ в алфавіті* X. *Говорять, що словникова функція* f *обчислюється ДМТ* M, *якщо для довільного слова* $p \in (X \setminus \{\#\})^*$ *має місце рівність* $M(p) = f(p)$. *Якщо така ДМТ існує, то функція* f *називається* **частково рекурсивною.**

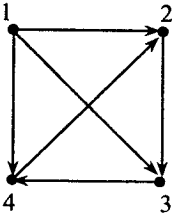
Приклад 4.2.4. a) Словникова функція $f: \{0, 1\}^* \to \{0, 1, \#\}^*$ така, що $f(p) = \#p$, є рекурсивною, оскільки для її обчислення існує ДМТ. Ця ДМТ була побудована в прикладі 4.2.1.

b) Рекурсивною є також функція $s(n) = n + 1$, оскільки ДМТ, що була побудована в прикладі 4.2.2, обчислює цю функцію.

c) Мова паліндромів у алфавіті X є рекурсивною, оскільки ДМТ з прикладу 4.2.3 розв'язує цю мову. ♠

Представлення проблем за допомогою слів. Слова в алфавіті $X \setminus \{\triangleright, \#\} = \{0, 1\}$, як зазначалося вище, служать для представлення

проблем. При такому підході має бути очевидним, що таке представлення проблеми у вигляді слова в алфавіті {0, 1} є достатньо загальним: довільний скінченний об'єкт може бути представлений словом скінченної довжини в цьому алфавіті. Наприклад, такий об'єкт, як скінченний граф, може бути записаний у вигляді слова скінченної довжини в алфавіті {0, 1}. Дійсно, використовуючи матрицю інцидентності, можна наведений нижче граф подати таким словом у цьому алфавіті:



«(1, 10), (1, 11), (1, 100), (10, 11), (11, 100), (100, 10)»
списки інцидентності

«(0111, 0010, 0001, 0100)»
матриця інцидентності

Як випливає з цього простого прикладу, цілі числа, скінченні множини, скінченні графи тощо можна представляти різними способами і такі представлення можуть відрізнятися один від другого як формою, так і довжиною. Але відома одна важлива властивість таких представлень:

Усі допустимі представлення-кодування є поліноміально еквівалентними.

Слова поліноміально еквівалентні означають, що якщо A і B два сенсовні представлення однієї й тієї проблем P і представлення A у вигляді слова має довжину n, то представлення B тієї самої проблеми P у вигляді слова має найбільшу довжину $p(n)$, де $p(n)$ — деякий поліном. Наприклад, представлення скінченного графа без ізольованих вершин у вигляді матриці інцидентності, вимагає найбільшої довжини слова, яка є квадратом довжини представлення цього графа у вигляді списків інцидентності.

Зауважимо, що теорія складності обчислень не залежить від проблеми представлення і її можна розв'язувати незалежно від слів і мов. Але розумний вибір представлення робить результати теорії складності адекватними реальним проблемам і практичній обчислювальності.

4.2.3. Багатострічкові машини Тьюрінга

Для означення часу і пам'яті, необхідних для оцінки складності даної проблеми, потрібне певне узагальнення ДМТ, а саме, багатострічкові машини Тьюрінга, тобто ДМТ з кількома стрічками. Це узагальнення, як буде показано далі, можна змоделювати за допомо-

гою звичайної однострічкової ДМТ. Отже, прийняття більшої кількості стрічок у моделі ДМТ не виводить нас за клас машин Тьюрінга.

Визначення 76. *ДМТ з k стрічками ($k \geq 1$) називається четвірка $M = (K, X, \delta, s_0)$, де K і X ті самі, що й в означенні звичайної однострічкової ДМТ, а функція переходів δ, яка називається програмою, визначає наступний стан таким чином:*

$$\delta: K \times X^k \to (K \cup \{h, «yes», «no»\}) \times (X \times \{l, r, t\})^k,$$

де

$$\delta(s, y_1, ..., y_k) = (s', z_1, d_1, ..., z_k, d_k)$$

означає, що коли ДМТ в деякий момент часу знаходиться в стані s, головка на першій стрічці оглядає символ y_1, і т.д., головка на k-й стрічці оглядає символ y_k, то в наступний момент часу ДМТ буде знаходитися в стані s', головка на першій стрічці запише символ z_1 замість символу y_1 і перейде або залишиться на місті залежно від значення d_1, і т.д., головка на k-й стрічці запише символ z_k замість символу y_k і перейде або залишиться в тій самій позиції, залежно від значення d_k.

На місце символу $\triangleright$ не дозволяється нічого записувати, тобто це означає, що коли $y_i = \triangleright$, то $d_i = r$. На початку на всіх стрічках записаний символ $\triangleright$, а на першій стрічці записано ще і вхідне слово p (тобто $q = \triangleright p$).

Результат роботи багатострічкової ДМТ визначається так само, як і для звичайної ДМТ, з тією лише різницею, що результат обчислень словникової функції після зупинки ДМТ записується на останній k-й стрічці.

Приклад 4.2.5. Паліндроми можна розв'язувати більш ефективно за допомогою двострічкової ДМТ, порівняно з однострічковою ДМТ із прикладу 4.2.3. Двострічкова ДМТ починає свою роботу з копіювання вхідного слова з першої стрічки на другу. Після цього ДМТ встановлює головку першої стрічки на перший символ вхідного слова, а головку другої стрічки — на останній символ скопійованого слова. А далі робота ДМТ зводиться до руху головок у протилежних одна одній напрямках і порівнянню на кожному кроці символів, що оглядаються головками. У випадку рівності цих символів, символ на другій стрічці замінюється пустим символом (тобто цей символ просто стирається на другій стрічці).

Таблицю функції переходів цієї ДМТ наведено нижче.

$k \in K$	$x \in X$	$y \in X$	$\delta(k, x, y)$
s_0	0	#	$(s_0, 0, r, 0, r)$
s_0	1	#	$(s_0, 1, r, 1, r)$
s_0	$\triangleright$	$\triangleright$	$(s_0, \triangleright, r, \triangleright, r)$
s_0	#	#	$(s, \#, l, \#, t)$
s	0	#	$(s, 0, l, \#, t)$
s	1	#	$(s, 1, l, \#, t)$
s	$\triangleright$	#	$(s_1, \triangleright, r, \#, l)$
s_1	0	0	$(s_1, 0, r, \#, l)$
s_1	1	1	$(s_1, 1, r, \#, l)$
s_1	0	1	$(«no», 0, t, 1, t)$
s_1	1	0	$(«no», 1, t, 0, t)$
s_1	#	$\triangleright$	$\{«yes», \#, t, \triangleright, r)$

Визначення 77. *Конфігурацією k-стрічкової ДМТ M називається кортеж вигляду* $(s, p_1, q_1,..., p_k, q_k)$, *де s — поточний стан ДМТ;* p_i, q_i *— слова, що записані на і-й стрічці* $(1 \le i \le k)$, *головка якої оглядає останній символ слова* p_i.

Говорять, що ДМТ M з конфігурації $(s, p_1, q_1,..., p_k, q_k)$ *безпосередньо досягає конфігурації* $(s', p_1', q_1',..., p_k', q_k')$ *(позначення* $(s, p_1, q_1,..., p_k, q_k) \xrightarrow{M} (s', p_1', q_1',..., p_k', q_k')$*), якщо виконуються такі умови. Нехай* $p_i = p_{i-1}y_i$, $q_i = xq_{i-1}$ *для* $i = 1, 2,...,$ *k і нехай*

$$\delta(s, y_1,..., y_k) = (s', z_1, d_1,..., z_k, d_k).$$

тоді для кожного $i = 1, 2,...,$ *k*

— якщо $d_i = r$, *то* $p_i' = p_{i-1}z_ix$, *а* $q_i' = q_{i-1}$;

— якщо $d_i = l$, *то* $p_i' = p_{i-1}$, *а* $q_i' = z_ixq_{i-1}$;

— якщо $d_i = t$, *то* $p_i' = p_{i-1}z_i$, *а* $q_i' = q_i$.

Іншими словами, для кожної стрічки повинні виконуватися умови переходу від однієї конфігурації до другої як для ДМТ з однією стрічкою.

Транзитивне замикання відношення безпосередньої досяжності для конфігурацій k-стрічкової ДМТ називається відношенням досяжності *для конфігурацій (позначення* $(s, p_1, q_1,..., p_k, q_k) \xrightarrow{M^*} (s', p_1', q_1',..., p_k', q_k')$*). Це означає, що існує таке число* $n \in N$, *що*

$$(s, p_1, q_1,..., p_k, q_k) \xrightarrow{M^n} (s', p_1', q_1',..., p_k', q_k'),$$

і в цьому випадку будемо говорити, що друга конфігурація досяжна з першої конфігурації за n кроків.

k-стрічкова ДМТ починає обчислення на вхідному слові p, знаходячись у конфігурації

$$(s_0, \triangleright, p, \triangleright, e, ..., \triangleright, e),$$

де e — пусте слово. Це означає, що вхідне слово p записується на першій стрічці, а на решті стрічок записаний тільки початковий символ $\triangleright$. Якщо

$$(s_0, \triangleright, p, \triangleright, e, ..., \triangleright, e) \xrightarrow{M^*} (\text{«}yes\text{»}, p_1, q_1, ..., p_k, q_k)$$

для деяких слів $p_1, q_1, \ldots, p_k, q_k$, то говоримо, що $M(p) = \text{«}yes\text{»}$. Якщо

$$(s_0, \triangleright, p, \triangleright, e, ..., \triangleright, e) \xrightarrow{M^*} (\text{«}no\text{»}, p_1, q_1, ..., p_k, q_k)$$

для деяких слів $p_1, q_1, ..., p_k, q_k$, то говоримо, що $M(p) = \text{«}no\text{»}$. Нарешті, якщо ДМТ зупиняється в конфігурації

$$(h, p_1, q_1, ..., p_k, q_k),$$

то $M(p) = p'_k q_k$, де p'_k збігається зі словом p_k, взятим без першого символу $\triangleright$, а q_k слово, яке записане без пустих символів. Це означає, що у випадку зупинки k-стрічкової ДМТ в стані h результат записується на останній k-й стрічці. Завдяки цьому звичайна однострічкова ДМТ є окремим випадком k-стрічкової ДМТ при $k = 1$. Крім того, це дозволяє визначити поняття обчислюваної словникової (рекурсивної) функції, розв'язуваності і розпізнавання мов аналогічно тому, як це визначалося для однострічкових ДМТ.

4.2.4. Поняття класу складності. Класи P, PSPACE, L і нижче

Далі нам потрібні будуть не довільні k-стрічкові ДМТ, а деякі спеціальні. Зокрема, для побудови класів складності використовуються 3-стрічкові ДМТ. Більш точно, 3-стрічкова ДМТ має три стрічки, серед яких

— на першій стрічці, яка називається **вхідною**, записується вхідне слово, і з цієї стрічки дозволяється лише читати і не дозволяється нічого писати;

— друга стрічка, яка називається **робочою** або **внутрішньою**, така, що на неї можна писати і з неї можна читати;

— третя стрічка, яка називається **вихідною**, така, що на неї дозволяється тільки писати і не дозволяється нічого читати.

Розв'язання проблеми такою машиною виконується так. На вхідній стрічці записується вхідне слово в алфавіті $X = \{0, 1, \triangleright, \#\}$ починаючи з лівої клітинки (тобто з першої клітинки стрічки), а в решту кліти-

нок стрічки записаний символ # — символ пустої клітинки. Машина працює з цим словом відповідно до своєї програми (функції переходів), записує необхідні символи на вихідну і робочі стрічки. Запис на вихідну стрічку теж виконується з першої лівої клітинки, а в решту клітинок стрічки записаний символ #. Якщо в деякий момент часу машина зупиняється з певною відповіддю, то вона записує цю відповідь — *«yes»* або *«no»* — на вихідну стрічку.

Означені вище поняття однострічкової і k-стрічкової ДМТ відносяться до поняття ДМТ, тобто відношення переходів у такій ДМТ є функціональним.

У теорії складності обчислень принциповий інтерес мають тільки два ресурси для базової моделі обчислень — *час* і *пам'ять*.

Визначення 78. *Якщо для 3-стрічкової ДМТ і вхідного слова p істинне відношення*

$$(s_0, \triangleright, p, \triangleright, e, \triangleright, e) \xrightarrow{M^t} (H, p_1, q_1, p_2, q_2, p_3, q_3),$$

де $H \in \{h, «yes», «no»\}$, то число t називається ***часом обчислень*** *або* ***часовою складністю*** *розв'язання проблеми. Іншими словами, часовою складністю розв'язання проблеми називається число кроків 3-стрічкової ДМТ, які вона виконала до зупинки.*
Пам'яттю *або* ***простором обчислень*** *називається число клітинок робочої стрічки, які використала ДМТ, під час обчислень.*

Зауважимо, що для ДМТ пам'ять, яка використовувалась під час обчислень, може бути набагато меншою, ніж розмір вхідного слова, а час обчислень повинен бути принаймні не меншим від довжини вхідного слова. Це пов'язано з тим, що відповідь часто залежить тільки від деякого початкового підслова вхідного слова, а не від усього слова.

Визначення 79. *Нехай $f: N \to N$ деяка функція. Говорять, що ДМТ M* **працює в часі** *$f(n)$, якщо для довільного вхідного слова p час обчислень, необхідний ДМТ M для розв'язання проблеми p, дорівнює $f(l(p))$, де $l(p) = n$ — довжина слова p. Функція f називається* **часовим обмеженням** *ДМТ M.*

Тепер можна ввести поняття класу складності, використовуючи 3-стрічкові ДМТ і розв'язувані ними мови.

Визначення 80. *Нехай $L \subseteq (X \cup \{\#\})^*$ — мова. Вважають, що мова L належить* **класу TIME($f(n)$)**, *якщо існує 3-стрічкова ДМТ,*

яка розв'язує мову L у часі $f(n)$. Множину мов **TIME**$(f(n))$ *називають* **класом часової складності.**

Якщо мова L розв'язується 3-стрічковою ДМТ в поліноміальному часі, тобто $f(n) = n^k$, де k — стала величина, то клас таких мов за всіма k складає **поліноміальний клас часової складності** *і позначається цей клас* **P**. *Отже,*

$$\mathbf{P} = \mathbf{TIME}(n^k) = \bigcup_{j>1} \mathbf{TIME}(n^j).$$

Говорять, що мова L належить до класу **SPACE**$(f(n))$, *якщо існує 3-стрічкова ДМТ, яка розв'язує мову L і використовує не більше $f(n)$ клітинок робочої стрічки. Множину мов* **SPACE**$(f(n))$ *називають* **класом складності по пам'яті.**

Якщо мова L розв'язується 3-стрічковою ДМТ при поліноміальній пам'яті, тобто $f(n) = n^k$, де k — стала величина, то клас таких мов за всіма k складає **поліноміальний клас складності по пам'яті** *і позначається цей клас* **PSPACE**, *тобто*

$$\mathbf{PSPACE} = \mathbf{SPACE}(n^k) = \bigcup_{j>1} \mathbf{SPACE}(n^j).$$

Говорять, що мова L належить класу **SPACE**$(\log n)$, *якщо існує 3-стрічкова ДМТ, яка розв'язує мову L і використовує не більше* $\log n$ *клітинок робочої стрічки. Множину мов* **SPACE**$(\log n)$ *називають* **класом логарифмічної складності по пам'яті** *і позначають через* **L** (*або іще* **LOGSPACE**), *тобто*

$$\mathbf{L} = \mathbf{SPACE}(\log n).$$

Зауважимо, що визначені вище класи складності **P**, **PSPACE**, **L** є найбільш важливими і популярними класами в теорії складності обчислень. Розглянемо приклади.

Приклад 4.2.6. Неважко показати, що ДМТ із прикладу 4.2.3 розпізнає паліндроми в часі, пропорціональному величині $f(n) = n^2$. Робота цієї ДМТ виконується у два етапи. На першому етапі за $2n + 1$ кроків виконується порівняння першого і останнього символів. Потім повторюється та сама робота над словом довжини $n - 2$, потім над словом довжини $n - 4$ і т.д. Отже, загальне число кроків ДМТ в найгіршому випадку буде $f(n) = \dfrac{(n+1)(n+2)}{2}$. Звідси випливає, що мова паліндромів належить до класу **TIME** $\dfrac{(n+1)(n+2)}{2} = O(n^2)$. Слід зазначити, що одержана оцінка є найбільш песимістичною, тому що

для слова 01^n ця ДМТ за $n + 3$ кроків визначає, що дане слово не є паліндромом. Але обчислюючи $f(n)$, необхідно брати до уваги найгірший можливий випадок вхідних даних.

Нарешті зазначимо, що 2-стрічкова ДМТ з прикладу 4.2.4 розв'язує мову паліндромів у часі $f(n) = 3n + 3 = O(n)$, звідки випливає, що ця мова належить класу складності **TIME**$(3n + 3)$. ♠

Приклад 4.2.7. Розглянемо таку 3-стрічкову ДМТ, яка розпізнає паліндроми. На першій стрічці цієї ДМТ записане вхідне слово p і вміст цієї стрічки далі не змінюється. Роботу ДМТ можна поділити на етапи. На i-му етапі на робочій стрічці МТ записане у двійковій системі числення число i, і МТ намагається розпізнати i-й символ слова p. Це виконується шляхом запису на третю стрічку числа $j = 1$ і подальшого порівняння i і j. Порівняння двох слів на стрічках є простою дією і виконується вона шляхом відповідних рухів головок.

Якщо $j < i$, то збільшуємо j на одиницю (ДМТ з прикладу 4.2.2) і пересуваємо головку на першій стрічці на одну клітинку вправо з метою аналізу наступного символу.

Якщо $i = j$, то запам'ятовуємо в стані символ вхідного слова, який оглядається головкою, встановлюємо знову $j = 1$ і шукаємо i-й символ від кінця слова p. Виконується це аналогічно до попереднього, але рух головки на першій стрічці виконується вліво, а не вправо, як раніше.

Якщо знайдені символи різні, то ДМТ зупиняється в стані *«no»*. Якщо ці символи рівні, то збільшуємо i на 1 і починаємо наступний $i + 1$-й етап. Нарешті, якщо під час деякого етапу i-й символ слова p виявиться рівним #, то це означає, що $i > n$ і, отже, вхідне слово p є паліндромом.

Скільки пам'яті необхідно цій ДМТ? Очевидно, ця пам'ять не перевищує величини O $(\log n)$. Машина тільки читає вхідне слово p і нічого не пише на першу стрічку, а слова, що записані на другій і третій стрічках, весь цей час не перевищують довжини $\log n$, де $n = l(p)$. ♠

Аналогічно до означення класів **P**, **PSPACE**, **L** визначаються і наведені нижче класи складності, які часто можна зустріти в застосуваннях. Ці класи лежать в ієрархії нижче від класу **P** і їх означення таке:

— **константа**: існує така константа k, що мова L розв'язується ДМТ за k кроків. Клас усіх таких мов позначається **TIME**(k);

— **логарифм**: мова L розв'язується ДМТ за $\log n$ кроків. Клас усіх таких мов позначається **TIME**$(\log n)$;

— **полілогарифм**: існує така константа k, що мова L розв'язується ДМТ за $\log^k n$ кроків. Клас усіх таких мов позначається **TIME**$(\log^k n)$;

— **лінійна**: мова L розв'язується ДМТ за n кроків. Клас усіх таких мов позначається TIME(n).

Таким же чином позначаються й класи складності по пам'яті.

4.2.5. Еквівалентність багатострічкових і однострічкових ДМТ

Покажемо тепер, що k-стрічкові ДМТ моделюються однострічковими ДМТ. Має місце така

Теорема 80. *Для довільної k-стрічкової ДМТ M, яка працює в часі $f(n)$, існує однострічкова ДМТ M', що працює в часі $O(f(n)^2)$, така, що для довільного слова $p \in F(X)$ має місце рівність $M(p) = M'(p)$.*

Доведення. Нехай $M = (K, X, \delta, s_0)$ k-стрічкова ДМТ. Побудуємо ДМТ $M' = (K', X', \delta', s_0)$, єдина стрічка якої моделює k стрічок машини M. Це можна зробити шляхом збереження на стрічці машини M' слово, яке є конкатенацією слів, записаних на стрічках ДМТ M (і яке не включає символів $\triangleright$, оскільки такі символи перешкоджали б рухам головки вздовж стрічки). Крім того, необхідно запам'ятати як положення кожної головки на кожній із k стрічок, так і кінець кожного із слів, які були записані на цих стрічках.

Для того, щоб реалізувати намічену вище конструкцію ДМТ M', приймемо такі позначення. Нехай $X' = X \cup \underline{X} \cup \{\triangleright', \triangleleft,\}$, де $\underline{X} = \{\underline{x} : x \in X\}$ є множиною версій символів, які оглядаються головкою ДМТ M'. Символ $\triangleright'$ — це нова версія початкового символу $\triangleright$, з якого можливий рух головки тільки вправо, а символ $\triangleleft$ означає правий кінець слова.

Довільну конфігурацію ДМТ M

$$(s, p_1, q_1, \ldots, p_k, q_k)$$

тепер можна змоделювати за допомогою такої конфігурації

$$(s, \triangleright\ p_1', q_1 \triangleleft \ldots,\ p_k', q_k \triangleleft\triangleleft),$$

де $p_i' = \triangleright' p_i$ і символ $\underline{x}_i$ слова p_i' оглядається головкою ДМТ M. Два символи, що йдуть підряд, $\triangleleft\triangleleft$ означають кінець слова на стрічці ДМТ M'.

Моделювання ДМТ M' має виконати пересування вхідного слова p на одну клітинку вправо, вставити символ $\triangleright'$ і після цього за останнім символом слова p вставити слово вигляду $\triangleleft\ (\triangleright'\ \triangleleft)^{k-1}\ \triangleleft$. Це можна зробити за допомогою введення у множину станів K' ДМТ

M' $2k + 2$ нових станів, єдиним призначенням яких є «пристосування» вхідного слова до потреб цієї ДМТ.

Машина M' виконує моделювання одного руху ДМТ M за допомогою двократного перегляду слова, яке записане на її стрічці, зліва направо і навпаки. При першому перегляді слова ДМТ M' збирає інформацію про k символів, які оглядаються головками ДМТ M, — ці символи підкреслені в цьому слові. Для запам'ятовування зібраної інформації ДМТ M' необхідно декілька нових станів — по одному стану для кожної комбінації стану ДМТ M і k-запису символів M.

На основі свого стану і в момент завершення першого перегляду вхідного слова ДМТ M' знає зміни, які необхідно виконати у слові, щоб змоделювати рух головки ДМТ M. Для цього M' переглядає свою стрічку ще раз справа наліво, зупиняючись на кожному підкресленому символі для того, щоб змінити один або два сусідніх символи так, щоб правильно «закодувати» запис символу і руху головки в тому напрямку, в якому вона пересувалася в ДМТ M. Ці зміни легко виконати за допомогою інформації, зібраної ДМТ M'. Але існує деяка складність, пов'язана з тим, що головка однієї зі стрічок оглядає останній символ і хоче пересунутися вправо. З цією метою необхідно вставити пусту клітинку на стрічці (тобто вставити клітинку і записати до неї пустий символ #). Для цього символ, що оглядається, $\triangleleft$ замінимо на символ $\triangleleft'$, пересунемо головку ДМТ M' у кінець слова (тобто до кінця вправо) аж до символів $\triangleleft\triangleleft$, а потім пересунемо всі символи на одну клітинку вправо за допомогою команд ДМТ з прикладу 4.2.1. Коли повторно дійдемо до символу $\triangleleft'$, то пересунемо його вправо на одну клітинку і замінимо на символ $\triangleleft$, а на його місце запишемо символ #. Далі переходимо до внесення змін у чергове слово ДМТ M.

Моделювання ДМТ M таким способом продовжується доти, поки M не зупиниться. У момент зупинки ДМТ M' стирає вміст усіх клітинок на своїй стрічці, крім вмісту клітинок, що відповідають останньому слову (для того щоб результат її роботи був таким самим, як і результат роботи ДМТ M), і зупиняється.

Скільки часу потребує моделювання роботи ДМТ M за допомогою ДМТ M'? Оскільки ДМТ M зупиняється в часі $f(l(p))$, то жодне із слів ДМТ M, що записані на її стрічках, не перевищує довжини $f(l(p))$ (це є дуже важливим і основним фактом, який стосується всіх розумних моделей обчислень: **ці моделі ніколи не можуть вимагати часу більшого, ніж пам'яті!**). Отже, загальна довжина слова на стрічці M' ніколи не перевищує величини $k(f(l(p) + 1) + 1$ (приймаючи до уваги також і символи $\triangleleft$). Моделювання одного кроку роботи ДМТ M потребує двократного перегляду слова зліва направо і на-

впаки, а це $4k(f(l(p)) + 1) + 4)$ рухів плюс найбільше $3k(f(l(p)) + 1) + 3)$ рухів на кожну стрічку ДМТ M, що моделюється. Остаточно одержуємо оцінку $O(k^2 f(l(p))^2)$ або $O(f(l(p))^2)$, оскільки величина k є сталою і не залежить від довжини $l(p)$. ■

З цієї теореми випливає, що збільшення числа стрічок не збільшує обчислювальної потужності багатострічкових ДМТ порівняно з однострічковими ДМТ. Єдине до чого приводить збільшення кількості стрічок, так це поліноміальне підвищення ефективності такої ДМТ.

4.2.6. Варіації машин Тьюрінга

4.2.7. Недетерміновані машини Тьюрінга. Класи NP, NSPACE, NL і EXP

Недетермінована машина Тьюрінга означається аналогічно ДМТ, тільки відношення переходів у такій МТ не є функцією, а є відношенням. Це означає, що множина можливих обчислень може розгалужуватися як дерево. Наведемо формальні означення.

Визначення 81. *Недетермінованою МТ (НДМТ) називається четвірка* $M = (K, X, \Delta, s_0)$, *де* K, X, s_0 *ті самі об'єкти, що і в ДМТ, а* Δ *— відношення переходів, яке визначає декілька можливих станів, у які МТ може перейти. Це означає, що НДМТ має можливість вибору між кількома способами і* Δ *вже не є функцією, тобто*

$$\Delta \subset (K \times X) \times ((K \cup \{h, «yes», «no»\}) \times X \times \{l, r, t\}).$$

Таким чином, для кожної комбінації стан і символ може існувати більше одного наступного допустимого стану або такого стану може не існувати жодного.

Конфігурацією НДМТ називається трійка (s, p', q'), де $s \in K$, а p' і q' — слова, відповідно зліва, включаючи і символ, що оглядається головкою, і праворуч від головки НДМТ. Говорять, що НДМТ переходить від конфігурації (s, p', q') до конфігурації (s', p'', g'') за один крок, якщо існує такий рух головки $((s, x), (s', y, d)) \in \Delta$, що

a) $d = r$ і слово p'' збігається зі словом p', в якому останній символ x замінено символом y і перший символ слова q' дописаний останнім символом до слова p'' (можливо, цей символ пустий #, якщо $q' = e$), а слово q'' збігається зі словом q', у якого стертий перший символ;

b) $d = l$ і слово p'' збігається зі словом p' без останнього символу x, який замінений символом y і цей символ y дописується першим символом до слова q'';

c) $d = t$ і слово p'' збігається зі словом p', в якому останній символ x замінено символом y і $q' = q''$.

Відношення $\underline{NM^k}$ і $\underline{NM^*}$ можна визначити так само, як і для ДМТ, але при цьому слід пам'ятати, що Δ є не функцією, а відношенням.

Визначення 82. *Нехай NM — деяка НДМТ і L — мова. Говорять, що NM* **розв'язує мову** *L, якщо для кожного $p \in X^*$ має місце наступне: $p \in L$ тоді і тільки тоді, коли*

$$(s_0, \triangleright, p) \overset{NM^*}{\to} (\text{«yes»}, p'q'),$$

де p', q' — деякі слова.

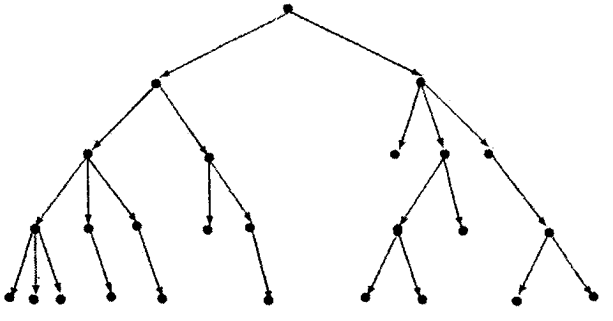
Це означення свідчить про те, що НДМТ мають певні переваги над іншими моделями обчислень. Вхідне слово p сприймається тоді і тільки тоді, коли існує деяка послідовність недетермінованих переходів, яка переводить НДМТ у стан *«yes»*. Інші послідовності переходів можуть не сприймати слово p, але для сприйняття p достатньо існування хоча б однієї такої послідовності. Слово p не сприймається тоді і тільки тоді, коли не існує жодної послідовності переходів, яка переводить НДМТ у стан *«yes»*.

Визначення 83. *НДМТ NM розв'язує мову L у часі $f(n)$ ($f: N \to N$), якщо NM розв'язує L і, крім того, для довільного $p \in X^*$ такого, що*

$$(s_0, \triangleright, p) \overset{NM^k}{\to} (\text{«yes»}, p', q'),$$

має місце нерівність $k \le f(l(p))$.

Із цього означення випливає, що *NM* розв'язує мову L у часі $f(l(p))$, якщо всі послідовності, що переводять *NM* у стан *«yes»*, не перевищують довжини $f(l(p))$. Усі послідовності обчислень *NM* можна подати у вигляді дерева:



Недетерміновані обчислення

Варто зауважити, що загальний час обчислень НДМТ може бути набагато більшим, ніж час, який вважається часом обчислень НДМТ (це легко помітити з наведеного вище рисунка).

Множина мов, що розв'язуються НДМТ, є новим важливим класом складності і позначається цей клас **NTIME**($f(n)$). Найважливішим класом складності є клас **NP**, який означається як теоретико-множинне об'єднання класів **NTIME**(n^k), де k — деяке стале число, тобто

$$\mathbf{NP} = \bigcup_{k>1} \mathbf{NTIME}(n^k).$$

Зауважимо, що клас **P** є підкласом класу **NP**, оскільки клас ДМТ є підкласом НДМТ. Іншими важливими класами складності є класи

$$\mathbf{NPSPACE} = \mathbf{NSPACE}(n^k) \text{ і } \mathbf{EXP} = \mathbf{TIME}(2^{n^k}).$$

Для класів складності по пам'яті можна розглядати класи складності нижче лінійної. Двома такими класами складності є вже відомий нам клас $\mathbf{L} = \mathbf{SPACE}(\log n)$ і клас $\mathbf{NL} = \mathbf{NSPACE}(\log n)$. Наведеними вище класами далеко не вичерпуються всі теоретично можливі класи складності. Зокрема, наведені нижче теореми свідчать про те, що таких класів існує досить багато. Для формулювання цих теорем уточнимо поняття функції складності, а саме, введемо поняття конструктивної функції складності.

Визначення 84. *Нехай $f: N \to N$ деяка функція натурального аргументу. Вважають, що функція f є конструктивною функцією складності (часу і пам'яті), якщо f є монотонною (тобто $\forall n \in N\, f(n+1) \geq f(n)$) і виконується така умова: існує k-стрічкова ДМТ $M_f(K, X, \delta, s_0)$, така що для довільного $n \in N$ і слова $p \in F(X)$ довжини n істинне відношення*

$$(s_0, \triangleright, p, \triangleright, e, ..., \triangleright, e) \overset{M_f}{\to} (h, p, \triangleright, \#^{j_1}, \triangleright, \#^{j_2}, ..., \triangleright, \sqcap^{f(l(p))},$$

де $\sqcap$ називається «квазіпустим» символом. Більше того, $t = O(n + f(n))$ і $j_i = O(f(l(p)))$ для $i = 2, ..., k-1$. Іншими словами, на слові p ДМТ M_f обчислює слово $\sqcap^{f(l(p))}$ і, більше того, для довільного слова p ДМТ M_f зупиняється після $O(l(p) + f(l(p)))$ кроків обчислень і використовує при цьому $O(f(l(p)))$ клітинок на робочих стрічках.

Мають місце такі твердження.

Теорема 81 (*Хартманіс і Стірнс*). *Якщо $F_1(n)$ і $F_2(n)$ є деякими конструктивними функціями часу і якщо*

$$\lim_{n\to\infty} \frac{F_1(n)\log_2(F_2(n))}{F_2(n)} = 0,$$

то існує мова L така, що L може бути розпізнана за допомогою деякої ДМТ у часі $F_2(n)$, але ніякою ДМТ в часі $F_1(n)$.

Теорема 82 (*Хартманіс, Левіс і Стірнс*). *Якщо $F_1(n)$ і $F_2(n)$ є деякими конструктивними функціями пам'яті і якщо*

$$\lim_{n\to\infty}\frac{F_1(n)}{F_2(n)}=0,$$

то існує мова L така, що L може бути розпізнана за допомогою деякої ДМТ при розмірах пам'яті, що обмежена $F_2(n)$, але ніякою ДМТ при розмірах пам'яті, обмеженої $F_1(n)$.

Теорема 83 (*Сейферас, Фішер, Мейер*). *Якщо $F_1(n)$ і $F_2(n)$ є деякими конструктивними функціями часу і якщо*

$$\lim_{n\to\infty}\frac{F_1(n)}{F_2(n)}=0,$$

то існує мова L така, що L може бути розпізнана за допомогою деякої НДМТ в часі $F_2(n)$, але ніякою НДМТ в часі $F_1(n)$.

Між ДМТ і НДМТ існує певний зв'язок, який виражається таким твердженням.

Теорема 84. *Нехай деяка НДМТ NM розв'язує мову L у часі $f(n)$, тоді існує 3-стрічкова ДМТ M, яка розв'язує мову L у часі $O(c^{f(n)})$, де c — константа, яка залежить від NM, тобто*

$$\mathbf{NTIME}\subseteq\bigcup_{c>1}\mathbf{TIME}(c^{f(n)}).$$

Доведення. Нехай $NM=(K, X, A, s_0)$ — деяка НДМТ. Для кожної пари $(s, x)\in K\times X$ розглянемо множину можливих виборів

$$C_{s,x}=\{(s', x', d):((s, x), (s', x', d)\in\Delta\}.$$

Очевидна скінченність такої множини, що дає можливість задати число $n=\max\limits_{s,x}|C_{s,x}|$, яке називається степенем недетермінізму *NM*. Припустимо, що $n>1$ (інакше *NM* буде детермінованою і тоді все доведено).

Основна схема моделювання роботи *NM* така. Довільне обчислення *NM* є послідовністю недетермінованих виборів. За прийнятих позначень довільна послідовність t недетермінованих виборів, що виконані *NM*, є послідовністю цілих чисел, які належать множині 0, 1,..., $n-1$. ДМТ *M* розглядає всі такі послідовності виборів у по-

рядку зростання їх довжини і моделює роботу *NM* на кожній з них. Варто звернути увагу на те, що неможливо зразу розглянути послідовність довжин $f(l(p))$, де p — вхідне слово, оскільки M повинна працювати не знаючи функції обмеження $f(n)$. Розглядаючи послідовність $(c_1, c_2,..., c_t)$ МТ M копіює її на своїй робочій (другій) стрічці. Потім M моделює операцію МТ *NM* так, як би це робила сама *NM*, у випадку вибору c_i на i-ому кроці обчислень. МТ M моделює таким чином t початкових кроків *NM*. Якщо такі вибори приводять *NM* до зупинки в стані *«yes»* (можливо і раніше, ніж через t кроків), то M також зупиняється в стані *«yes»*. У протилежному випадку, M переходить до аналізу чергової послідовності виборів. Генерація чергової послідовності виконується просто: це відбувається так само, як обчислення чергового числа в n-ковій системі числення (див. приклад 4.2.2).

Яким чином МТ M визначає, що *NM* не сприймає слово p, не знаючи функції обмеження $f(n)$? Розв'язок тут дуже простий. Якщо M виконає операції *NM* на всіх послідовностях певної довжини t і не знайде жодного обчислення, яке може бути продовжено далі (це означає, що всі обчислення довжини t закінчуються або в стані h або в стані *«no»*), то M сигналізує про те, що *NM* не сприймає вхідне слово.

Час, необхідний ДМТ M на виконання моделювання роботи НДМТ *NM*, обмежений зверху числом

$$\sum_{t=1}^{f(n)} n^t = O\left(n^{f(n)+1}\right),$$

яке складає число послідовностей виборів, через які потрібно пройти. Це час, необхідний для генерації і розгляду кожного вибору чергового кроку обчислень. Неважко помітити, що одержана оцінка має тип $O(2^{f(n)})$. ■

Тепер можна уточнити поняття обмеженості по пам'яті для 3-стрічкової НДМТ $NM = (K, X, \Delta, s_0)$.

Визначення 85. *Говорять, що НДМТ NM розв'язує мову L у пам'яті* $f(n)$, *якщо NM розв'язує мову L і для довільного* $p \in (X \setminus \{\#\})^*$, *для якого*

$$(s_0, \triangleright, p, \triangleright, e, \triangleright, e) \overset{NM}{\to} (s, p_1, q_1, p_2, q_2, p_3, q_3),$$

має місце нерівність $l(p_2q_2) \le f(l(p))$.

Це означає, що НДМТ *NM* при обчисленнях жодного разу не потребує клітинок на її робочій стрічці більше, ніж значення функції f на довжині вхідного слова p. Зауважимо, що в даному означенні на-

віть не вимагається, щоб усі обчислення НДМТ були скінченними. Але все ж таки, для означення деяких типів проблем і алгоритмів за допомогою НДМТ будемо припускатися того, що її дерево досяжних конфігурацій є скінченним і включає всі фінальні конфігурації, тобто конфігурації, в яких НДМТ зупиняється, причому глибина таких конфігурацій така сама, як і глибина самого дерева. У цьому випадку просторомпам'яттю, що використовується, є максимальне число клітинок на робочій стрічці, що відвідуються, за всіма конфігураціями цього дерева.

На завершення даного підрозділу зазначимо, що недетермінізм займає особливе місце не тільки в теорії складності обчислень, а і в таких областях як штучний інтелект, оптимізація функцій і, особливо, у конструктивній логіці. Логіка, як правило, займається пошуком доведення тверджень, а цей пошук і складає основу недетермінізму. При доведенні тверджень у логіці шукаються різні можливі шляхи доведення твердження і для його доведення достатньо знайти хоча б один шлях, який веде до успіху. Аналогічна ситуація має місце і при оптимізації функцій, і в штучному інтелекті.

4.2.8. Альтернуючі машини Тьюрінга. Класи AP і AL

Існує і узагальнення поняття недетермінізму, яке називається **альтернацією**. Розглянемо спочатку нове означення недетермінізму, яке використовує поняття конфігурації. Конфігурація приводить до сприйняття слова тоді і тільки тоді, коли вона є фінальною і сприймаючою або принаймні одна з її конфігурацій приводить до сприйняття слова. Це означає, що в дійсності кожна конфігурація є альтернативою (*OR*) конфігурацій, які йдуть за нею в дереві обчислень (конфігурації, які йдуть за даною конфігурацією, називаються нащадками даної конфігурації). З другого боку, НДМТ, яка розв'язує доповнення тієї самої мови, мала б конфігурації, які були б кон'юнкціями (*AND*) конфігурацій нащадків.

Припустимо, що дана 3-стрічкова НДМТ працює в обох режимах. Це означає, що деякі конфігурації НДМТ є кон'юнкціями конфігурацій і сприймають вхідне слово тоді і тільки тоді, коли всі їх нащадки сприймають це слово, у той час як інші конфігурації є диз'юнкціями конфігурацій і сприймають вхідне слово тоді і тільки тоді, коли принаймні одна з їх конфігурацій нащадків сприймає це слово. Режим кожної конфігурації (*AND* або *OR*) визначається через стан конфігурації. НДМТ сприймає вхідне слово тоді і тільки тоді, коли її початкова конфігурація сприймає це слово. Наведемо формальні означення.

Визначення 86. *Альтернуючою МТ (АМТ) є НДМТ* $NM = (K, X, \Delta, s_0)$, *у якій множина станів розбита на дві підмножини* $K = K_{AND} \cup K_{OR}$. *Нехай* $p \in X^*$ — *вхідне слово. Розглянемо дерево обчислень NM для слова p. Кожна вершина дерева обчислень включає конфігурацію НДМТ і номер кроку обчислень. Означимо індуктивно підмножину конфігурацій, які будемо називати* **остаточно сприймаючими**. *З цією метою будемо рухатись по вершинах цього дерева знизу вверх, починаючи з його листків. Конфігурація C зі станом* s_{AND} *є остаточно сприймаючою тоді і тільки тоді, коли всі її конфігурації-нащадки є остаточно сприймаючими (конфігурація C', яка досяжна з конфігурації C за один крок обчислень, називається безпосередньою конфігурацією-нащадком C). Конфігурація C зі станом* s_{OR} *є* **остаточно сприймаючою** *тоді і тільки тоді, коли принаймні одна з її конфігурацій-нащадків є остаточно сприймаючою. Нарешті, АМТ NM сприймає вхідне слово p, якщо її початкова конфігурація є остаточно сприймаючою.*

Будемо говорити, що АМТ NM розв'язує мову L, якщо NM сприймає всі слова $p \in L$ *і не сприймає всі слова* $p \notin L$.

Через $\mathbf{ATIME}(f(n))$ — *альтернуючий час* $f(n)$ — *позначається клас усіх мов, які розв'язуються за допомогою АМТ в часі* $f(n)$, *де* $n = l(p)$, $p \in X^*$.

Через $\mathbf{ASPACE}(f(n))$ — *альтернуюча пам'ять* $f(n)$ — *позначається клас усіх мов, які розв'язуються за допомогою АМТ і при цьому АМТ використовує не більше ніж* $f(n)$ *пам'яті при обчисленнях на слові* $p \in X^*$, *де* $n = l(p)$.

Нарешті, означимо наступні класи складності

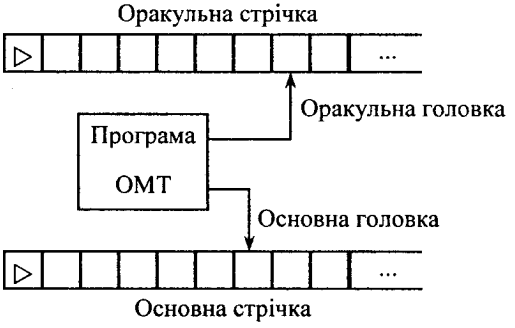
$$\mathbf{AP} = \mathbf{ATIME}(n^k) \text{ і } \mathbf{AL} = \mathbf{ASPACE}(\log n).$$

Як випливає з цього означення, ресурси часу і пам'яті визначаються так само, як і для НДМТ, але тепер з'являється додатковий ресурс, який називається «альтернативами». Це буде максимум за всіма шляхами з початкової конфігурації до фінальної конфігурації числа раз, коли конфігурація має позначку, відмінну від її батька (корінь не має батьків, тому він приймається за першу альтернативу). Отже, НДМТ може розглядатися як АМТ з однією альтернативою і всіма конфігураціями, які позначені як *OR*.

4.2.9. Оракульні машини Тьюрінга (ОМТ)

Існують ДМТ з додатковою нескінченною в один бік стрічкою, що називається **стрічкою-оракулом**, яка залежно від деяких умов може бути доступна тільки для читання або тільки для запису. Такі

машини називаються оракульними машинами Тьюрінга (ОМТ). Схему ОМТ наведено нижче на рисунку:



Оракульна машина Тьюрінга

Нехай $M = (K, X, \delta, s_0)$ — оракульна МТ. Оскільки ОМТ дуже схожа на ДМТ, опишемо тільки відмінності між ОМТ і ДМТ. Перша відмінність полягає в тому, що у множині станів ОМТ, крім станів h і s_0, виділяються ще два стани s_c і s_r, які називаються відповідно **станами запиту до оракула** і **резюмуючим станом**. Друга і головна відмінність полягає в тому, що ОМТ має можливість «консультуватися з оракулом». Для пояснення цієї відмінності опишемо роботу ОМТ.

Неформально робота ОМТ на вхідному слові $p \in X^*$ схожа на роботу ДМТ, за винятком того, що коли ОМТ знаходиться у стані s_c запиту до оракула, то перехід у наступний стан залежить від конкретної оракульної функції g: $X^* \to X^*$. Доповнимо цей неформальний опис таким формальним описом.

У початковий момент часу символи вхідного слова записані на основній стрічці ОМТ, головка ОМТ знаходиться в початковому стані s_0 і оглядає початковий символ ▷, на оракульній стрічці записані тільки початковий символ ▷, а всі інші символи пусті. Обчислення виконуються крок за кроком, причому на кожному кроці може виникнути одна із таких ситуацій:

a) Якщо поточний стан є заключним станом h, то обчислення закінчуються і робота ОМТ зупиняється.

b) Якщо поточним станом є стан $s \in K \setminus \{h, s_c\}$, то поведінка ОМТ залежить від символів, що прочитані на стрічках, і від функції переходів δ. Нехай x_1 — це символ у клітинці, що оглядається головкою на основній стрічці, x_2 — символ, що оглядається оракульною головкою в клітинці оракульної стрічки, і $(s', x_1', x_2', d_1, d_2)$ — значення функції переходів $\delta(s, x_1, x_2)$. Тоді ОМТ переходить зі стану s у стан

s'; головка на основній стрічці записує символ x_1' на місце символу x_1 і пересувається відповідно до значення d_1; оракульна головка записує символ x_2' на місце символу x_2 на оракульній стрічці і пересувається відповідно до значення d_2. Таким чином, усе відбувається як на одному кроці обчислень 2-стрічкової машини Тьюрінга.

c) Якщо поточний стан є станом запиту до оракула s_c, то подальший стан ОМТ залежить від вмісту оракульної стрічки і від оракульної функції g. Нехай $q \in X^*$ — слово, що записане на оракульній стрічці, причому оракульна головка оглядає останній символ цього слова. Тоді за один крок обчислень оракульна стрічка змінюється так, що в перших $l(z)$ її клітинках виявляється записаним слово $z = g(q)$; оракульна головка переходить на першу клітинку для перегляду цього слова і машина переходить із стану s_c у стан s_r. Цей крок не змінює вмісту основної стрічки і положення основної головки.

Таким чином, головна відмінність ОМТ від ДМТ проявляється в останній ситуації. Якщо ОМТ записує питання q на оракульну стрічку і потім приймає стан запиту до оракула, то слово-відповідь $z = g(q)$ видається за один крок обчислень. Це відповідає зверненню до гіпотетичної підпрограми обчислення функції g і обчислення ОМТ M на вхідному слові p залежить як від p, так і від оракульної функції g, що використовується.

Час і пам'ять, необхідні для роботи ОМТ, визначаються так само, як і для ДМТ. Щоправда, існує певна технічна проблема: чи потрібно враховувати число клітинок стрічки-оракула? Далі, якщо не обумовлене супротивне, буде вважатися, що простір стрічки-оракула враховується в загальному просторі, що використовується. Але один додатковий ресурс при цьому виникає — це кількість запитів до оракула, тобто максимум за всіма можливими способами обчислень кількості входжень у стан запиту.

4.2.10. Машини вільного доступу (RAM)

Писати програми для машин Тьюрінга досить складна і малоприємна справа. Ця модель обчислень була вибрана як основна, тому що вона є універсальною моделлю обчислень і за її допомогою можна моделювати без суттєвої втрати ефективності більш реальні моделі, такі як машини вільного доступу або *RAM* машини (*RAM* від англійського Random Access Mashine) [1, 55]. Писати програми для *RAM* машин легше, ніж для машини Тьюрінга, оскільки модель *RAM* максимально наближена до дійсності. Розглянемо поняття *RAM* машини більш детально.

RAM машини є моделями обчислювальних пристроїв і так само, як і МТ, складаються із програми і працюють з єдиною структурою, яка складається із **таблиці регістрів**. Кожний регістр може містити довільне як завгодно велике ціле число, як додатне, так і від'ємне. Команди *RAM* нагадують команди реальних комп'ютерів і тому *RAM* представляють найбільш близькі до реальних комп'ютерів моделі обчислень. Наведемо формальні означення.

Визначення 87. *Програмою RAM називається скінченна послідовність* $\Pi = (\pi_1,..., \pi_m)$, *елементи якої називаються командами, а кожна з команд належить до одного з перелічених нижче типів, наведених у таблиці на рис.* 4.2.1:

Команда	Аргумент	Значення
READ	J	$r_0 := i_j$
READ	$\uparrow j$	$r_0 := i_{r_j}$
STORE	J	$r_j := r_0$
STORE	$\uparrow j$	$r_{i_j} := r_0$
LOAD	X	$r_0 := x$
ADD	X	$r_0 := r_0 + x$
SUB	X	$r_0 := r_0 - x$
HALF		$r_0 := \left[\frac{r_0}{2}\right]$
JUMP	J	$k := j$
JPOS	J	*if* $r_0 > 0$ *then* $k := j$
JZERO	J	*if* $r_0 = 0$ *then* $k := j$
JNEG	J	*if* $r_0 < 0$ *then* $k := j$
JHALT		$k := 0$

Рис. 4.2.1. Типи команд RAM машини

У кожний момент часу обчислень *RAM* регістр i ($i > 0$) містить ціле число, яке може бути від'ємним і яке позначається через r_i. На кожному кроці обчислень виконується команда π_k, де k — лічильник програми *RAM*, що містить номер команди, яка виконується. Виконання команди може змінити вміст одного із регістрів і значення лічильника k. **Семантика команд** означена в таблиці з рис. 4.2.1 і, як видно з таблиці, ця семантика близька до семантики команд реального комп'ютера. Аргументами команд виступає ціле число j, r_j озна-

чає вміст j-го регістра, i_j є j-м вхідним елементом, x означає аргумент типу j, «$\uparrow j$» або «$=j$». Команди *READ* і *STORE* не можуть мати аргументів типу «$=j$». Значення j позначається r_j, аргумент «$\uparrow j$» має значення r_{r_j}, а «$=j$» — значення j. Символ k означає лічильник команд. Кожна команда, в якій немає явної зміни k, приводить до зміни значення k на $k + 1$.

Регістр 0 відіграє особливу роль і називається **акумулятором**. У цьому регістрі виконуються всі арифметичні і логічні операції. Кожна *RAM* має три способи адресації. Деякі з операцій використовують як аргумент x, де x може бути одним із трьох типів: якщо x є цілим числом j, то x означає регістр j, якщо x має тип $\uparrow j$, то це означає значення регістра, номер якого зберігається в j-му регістрі, якщо x має вигляд $=j$, то це означає саме число j.

Виконання команди приводить до зміни значення k. Усі перелічені команди в таблиці з рис. 4.2.1, за винятком останніх п'яти, змінюють значення лічильника з k на $k + 1$. Це означає, що наступною командою, яка повинна виконуватись у програмі *RAM*, буде наступна команда за даною командою в її програмі. Чотири команди переходу змінюють значення k лічильника і ця зміна часто залежить від вмісту акумулятора. Нарешті, команда *HALT* приводить до зупинки обчислень. Вважається, що довільна неправильна команда (наприклад, команда звернення до регістра з від'ємним номером) призводить до зупинки *RAM*, тобто до команди *HALT*.

На початку роботи програми *RAM* в усі її регістри записано значення 0. **Даними програми** *RAM* є скінченна послідовність чисел, які записані в скінченній таблиці регістрів даних $I = (i_1,..., i_n)$. Довільне число з цих вхідних даних може бути записане в акумулятор за допомогою команди *READ*. У момент закінчення обчислень результат цих обчислень записується в акумуляторі.

RAM виконує першу команду і проводить продиктовані цією командою зміни. Після цього *RAM* виконує команду π_k, де лічильник команд k приймає нове значення і так далі аж до зупинки машини. Сформулюємо більш точно поняття обчислень, що виконуються *RAM*, використовуючи, як і раніше, поняття конфігурації.

Визначення 88. **Конфігурацією машини RAM** *називається пара* $C = (k, R)$, *де* k *— номер команди, яка повинна бути виконана в даний момент, а* $R = \{(j_1, r_{j_1}), (j_2, r_{j_2}), ..., (j_s, r_{j_s})\}$ *— скінченна множина пар типу регістр-значення. Неформально множина* R *описує поточні значення регістрів, які одержані в результаті виконаних команд до даного моменту часу (нагадаємо, що в решту регістрів записані нулі). Конфігурація* $C = \{(1, \varnothing)\}$ *називається* **початковою**.

Нехай П — програма *RAM*, $I = (i_1,..., i_s)$ — вхідні дані, а $C = (k, R)$ і $C' = (k', R')$ — деякі конфігурації. Говорять, що *RAM* переходить від конфігурації C до конфігурації C' за один крок обчислень, що позначається $(k, R) \xrightarrow{\text{П}.I} (k'R')$, якщо мають місце такі залежності:

— число k' є новим значенням лічильника команд після виконання π_k — k-ї команди програми П;

— якщо команда з номером k є однією з п'яти останніх команд у таблиці з рис. 4.2.1, то $R = R'$;

— у решті випадках деякий регістр j має нове значення x', яке обчислене у відповідності з командою π_k.

Для того щоб отримати R', вилучаємо із R пару (j, x), якщо вона там є, і заносимо пару (j, x'). Тепер можна визначити відношення $(k, R) \xrightarrow{\text{П}.I^m} (k'R')$ і $(k, R) \xrightarrow{\text{П}.I^*} (k'R')$ як досяжність конфігурації за m кроків і транзитивне замикання відношення $\xrightarrow{\text{П}.I}$ відповідно.

Нехай П — програма *RAM*, D — множина скінченних послідовностей цілих чисел і нехай ϕ — цілочисельна функція на D. Вважають, що програма П обчислює функцію ϕ, якщо для довільного $I \in D$ має місце відношення $(I, \varnothing) \xrightarrow{\text{П}.I^*} (0, R)$, де $(0, \phi(I)) \in R$.

Розглянемо тепер означення часових вимог. Далі будемо припускати, що одна арифметична операція виконується за один крок. Це може здатися досить сильним обмеженням і викривленням оцінки, тому що, наприклад, виконання операції *ADD* для великих чисел буде коштувати принаймні логарифм від розміру її аргументів. Але вибрана в цій моделі *RAM* множина команд приводить до несуттєвого викривлення оцінки (зокрема, серед цих операцій немає операції множення цілих чисел). Математичним обґрунтуванням такого висновку є той факт (що доводиться нижче), що програма *RAM* з таким припущенням про час може бути змодельована за допомогою відповідної ДМТ тільки з поліноміальною величиною втрати часу.

Логарифм все ж таки потрібен для того, щоб визначити розмір вхідних даних. Нехай $b(i)$ буде двійковим представленням цілого числа i без непотрібних нулів на початку і зі знаком мінус, якщо i від'ємне. Довжиною цілого числа i будемо називати довжину його двійкового представлення $l(i) = l(b(i))$. Якщо $I = (i_1,..., i_s)$ — послідовність цілих чисел, то її довжиною називається число $n = l(I) = \sum_{j=1}^{s} l(i_j)$.

Припустимо тепер, що програма П машини *RAM* обчислює цілочисельну функцію ϕ на D.

Визначення 89. *Нехай* $f: N \to N$ *— цілочисельна функція натурального аргументу і нехай для довільного* $I \in D^s$ *має місце відношення* $(1, \varnothing) \xrightarrow{\text{П},t'} (0, R)$, *де* $t \le f(l(I))$, *тоді говорять, що програма* П *обчислює функцію* ϕ *в часі* $f(n)$. *Іншими словами, час обчислень RAM виражається як функція від сумарної довжини даних.*

Приклад 4.2.8. Розглянемо програму *RAM*, яка обчислює функцію (операцію) множення двох натуральних чисел, тобто функцію

$$\phi : N^2 \to N,$$

таку, що $\phi(i_1, i_2) = i_1 \cdot i_2$. Обчислення цієї функції виконується шляхом множення відповідних двійкових чисел, за якого команда *HALF* використовується для знаходження двійкового представлення i_2 (саме з цією метою зазначена команда була введена у множину команд *RAM*). Програма *RAM* має вигляд:

1	READ 1	(Регістр 1 містить i_1, а під час
2	STORE 1	k-ї ітерації 5-й регістр містить
3	STORE 5	$i_1 \cdot 2^k$. Спочатку $k = 0$)
4	READ 2	
5	STORE 2	(регістр 2 містить i_2)
6	HALF	(збільшуємо значення k і починаємо k-у ітерацію)
7	STORE 3	(регістр 3 містить цілу частину $[i_2/2^k]$)
8	ADD 3	
9	SUB 2	
10	JZERO 14	
11	LOAD 4	(додати значення рег. 5 і 4 стільки разів,
12	ADD 5	скільки k-й біт дорівнює 0, рахуючи справа)
13	STORE 4	(регістр 4 містить $i_2 \cdot (i_2 \bmod 2^k)$)
14	LOAD 5	
15	ADD 5	
16	STORE 5	(див. коментар до кроку 3)
17	LOAD 3	(регістр 2 містить i_2)
18	JZERO 20	(якщо $[i_2 / 2^k] = 0$, то закінчити роботу)
19	JUMP 5	(якщо ні, то повторюємо ітерацію)
20	LOAD 4	(результат)
21	HALT	

Рис. 4.2.2. Програма *MPLY* множення натуральних чисел для *RAM*

Досить уважно придивитися до цієї програми, щоб помітити, що в програмі команди, які знаходяться в циклі 5.—19., повторюються $[\log i_2]$ разів. Спочатку на k-й ітерації (а починаємо з ітерації 0) регістр 3 містить $[i_2/2^k]$, регістр 5 містить $i_1 \cdot 2^k$, а регістр 4 — $i_1 \cdot (i_2 \ mod \ 2^k)$. На кожній ітерації зберігається інваріант $i_1 \cdot (i_2 \ mod \ 2^k)$, і по закінченні ітерації перевіряється умова на рівність 0 вмісту регістра 3. Якщо цей вміст дорівнює 0, то закінчуємо обчислення і отримуємо результат на регістрі 4, інакше повторюємо чергову ітерацію.

Легко підрахувати, що наведена програма *MPLY* обчислює функцію $\phi(i_1, i_2) = i_1 \cdot i_2$ у часі $O(n)$. Нагадаємо, що під часом $O(n)$ розуміємо ціле число команд, яке пропорціональне логарифму довжини даних. Число ітерацій програми *MPLY* на рис. 4.2.2 фактично задовольняє умову $\log i_2 \le l(I)$, а кожна ітерація складається зі сталого числа команд.

Нарешті зазначимо, що коли програма *MPLY* вже існує, то можна в програмах *RAM* використовувати команди *MPLY x*, де x є одним з аргументів (j, $\uparrow j$, $=j$). У разі необхідності використання програми *MPLY* може бути реалізоване за допомогою команд, які складають цю програму (можливо на іншій множині регістрів). Реалізація операції множення від'ємних чисел вимагає декількох додаткових команд. ♠

Розглянемо тепер спосіб моделювання машини Тьюрінга за допомогою програми деякої *RAM*. Нехай $X = \{x_1, ..., x_k\}$ — алфавіт машини Тьюрінга, D_X — множина скінченних послідовностей цілих чисел, тобто $D_X = \{(i_1, ..., i_n, 0) : n \ge 0, 1 \le i_j \le k, j = 1, ..., n\}$. Якщо $L \in (X \setminus \{\#\})^*$ — деяка мова, то визначимо функцію $\phi_L : D_X \to \{0, 1\}$ таку, що $\phi_L(i_1, \ldots, i_n, 0) = 1$, якщо $x_{i_1} \ldots x_{i_n} \in L$, і 0 — у протилежному випадку. Останній 0 у послідовності даних програми *RAM* дозволяє цій програмі знайти кінець вхідного слова МТ.

Теорема 85. *Нехай L — мова, що належить до класу часової складності* TIME($f(n)$), *тоді існує програма RAM, яка обчислює функцію ϕ_L у часі $O(f(n))$.*

Доведення. Нехай (K, X, δ, s_0) — МТ, що розв'язує мову L у часі $f(n)$. Програма *RAM* починає свою роботу з копіювання даних у регістри від 4 до $n + 3$ (регістри 1 і 2 будуть потрібні для моделювання, а регістр 3 містить число j таке, що $x_j = \triangleright$). До регістру 1 заноситься значення 4, яке буде змінюватися під час моделювання, і служить для визначення положення головки МТ.

Тепер починаємо крок за кроком моделювати роботу МТ. Програма *RAM* має послідовність команд, що моделюють роботу МТ в кожному стані $s \in K$. Ця послідовність складається із $|X|$ підпослідов-

ностей. Будемо вважати, що j-а підпослідовність починається з команди з номером N_{s,x_j}. Припустимо, що $\delta(s, x_j) = (s', x_l, d)$. Команди j-ї підпослідовності мають такий вигляд:

N_{s,x_j}.	LOAD ↑ 1	(взяти символ, що оглядається головкою)
N_{s,x_j} + 1.	SUB j	(відняти j — номер символу, що оглядається)
N_{s,x_j} + 2.	JZERO N_{s,x_j} + 4	(якщо символ x_j, то зробити запис)
N_{s,x_j} + 3.	JUMP N_{s,x_j} + 1	(інакше перейти до чергового символу)
N_{s,x_j} + 4.	LOAD =1	(потрібно записати символ x_l)
N_{s,x_j} + 5.	STORE ↑ 1	(записали x_l)
N_{s,x_j} + 6.	LOAD 1	(позиція головки)
N_{s,x_j} + 7.	ADD = d	(1 для $d = r$, −1 для $d = l$ і 0 для $d = t$)
N_{s,x_j} + 8.	STORE 1	
N_{s,x_j} + 9.	JUMP N_{s',x_1}	(почати моделювання стану s')

Нарешті, стани *«yes»* і *«no»* моделюються за допомогою простої послідовності команд *RAM*, які записують константу 1 (відповідно 0) в акумулятор і виконують команду *HALT*.

Час, необхідний на моделювання команд виконання одного руху МТ, є сталим (залежить тільки від M). Додаткові $O(n)$ команд потрібні для копіювання даних. Цим завершується доведення. ■

Має місце і обернене твердження про те, що довільна програма *RAM* може бути змодельована за допомогою відповідної машини Тьюрінга з втратою ефективності, оцінка якої виражається поліномом. Для виконання цього моделювання визначимося з входом і виходом, для того щоб МТ могла працювати як програма *RAM*.

Нехай ϕ: $D \to D$ — деяка функція, що визначена на множині D усіх скінченних послідовностей цілих чисел. Двійкове представлення послідовності $I = (i_1,..., i_n)$, яке позначається через $b(I)$, є словом $b(i_1);...; b(i_n)$.

Визначення 90. *Говорять, що ДМТ M обчислює функцію ϕ, якщо для довільного $I \in D$ має місце рівність $M(b(I)) = b(\phi(I))$.*

Теорема 86. *Якщо програма RAM обчислює функцію* ϕ *в часі* $f(n)$, *то існує 7-стрічкова ДМТ M, яка обчислює* ϕ *в часі* $O(f(n)^3)$.

Доведення. Перша стрічка машини Тьюрінга M є вхідною і застосовується тільки для читання. На другій стрічці записані представлення значень регістрів R, які складаються із послідовності слів вигляду $b(i) : b(r_i)$ розділених знаком ';', а також, можливо, і із послідовності пустих символів. Усе слово в цілому закінчується символом $\triangleleft$. Щоразу, коли активізується значення регістра i (у тому числі й акумулятора), попередня пара $b(i) : b(r_i)$ стирається (на її місце записується пустий символ #), а нова пара записується в кінець слова.

Стани машини M розбиті на m груп, де m — число команд у програмі Ф. Потрібні в даний момент значення регістрів відшукуються шляхом перегляду другої стрічки зліва направо. Для цього необхідно не більше двох переглядів цієї стрічки. Третя стрічка містить значення лічильника k. Четверта стрічка містить адресу активного регістра. Щоразу, коли читається пара $b(i) : b(r_i)$, значення четвертої стрічки порівнюється з $b(i)$. Якщо ці значення рівні, то $b(r_i)$ переносяться на інші стрічки для подальших перетворень. Усі останні операції виконуються на тих трьох стрічках, що залишилися. У випадку трьох арифметичних операцій дві стрічки служать для зберігання аргументів, а результат записується на третій стрічці. Після обчислень активізується друга стрічка (стираємо одну пару, а другу дописуємо в кінець слова). Закінчується моделювання команди шляхом переходу до наступного стану і моделювання чергової команди відповідно до програми Ф. Коли дійдемо до команди *HALT*, то значення регістра 0 переписуємо з другої стрічки на сьому — вихідну стрічку.

Підрахуємо час, необхідний на моделювання таким способом $f(n)$ кроків програми Ф на даних I довжини n. Для цього спочатку доведемо такий корисний факт: після t кроків обчислень програми *RAM* на даних I вміст кожного регістра має довжину, яка не перевищує величини $t + l(I) + l(B)$, де B — найбільше ціле число явно записане в командах програми Ф.

Для доведення застосуємо метод математичної індукції за числом t. Базис індукції при $t = 0$, очевидно, має місце. Припустимо, що цей факт має місце після виконання $t - 1$ кроків обчислень. При виконанні кроку t можливі декілька випадків залежно від типу команди. Якщо команда, що виконується, є командою переходу-скачка або командою *HALT*, то ці команди не змінюють вмісту жодного із регістрів. Якщо такою командою є *LOAD* або *STORE*, то вмісти регістрів, що модифікуються, беруться з других регістрів з попередніх

кроків і, отже, факт має місце і в цьому випадку. Якщо такою командою є команда *READ*, то складова $l(I)$ гарантує справедливість факту, що доводиться. Нарешті, припустимо, що командою є одна з арифметичних команд, наприклад *ADD*, яка виконує додавання двох чисел i і j. Кожне з цих чисел або знаходиться в одному з регістрів і записано на одному з попередніх кроків, або записано дійсно в програмі Ф і не перевищує B. Довжина результату не перевищує величини $t - 1 + l(I) + l(B)$. Це означає, що даний факт має місце для операції *ADD*. Для операцій *SUB* викладки такі самі, як і у випадку *ADD*, а для команди *HALF* навіть дещо простіші (оскільки результат коротший від аргументу). Цим доведення даного факту закінчується.

Перейдемо тепер до оцінки часових обмежень. Необхідно показати, що моделювання команд програми Ф машини M потребує часу $O(f(n)^2)$, де n — довжина даних. Кодування поточної команди програми Ф і констант, що в ній знаходяться, можна виконати за константу. Читання з другої стрічки значень регістрів, які необхідні для виконання команди, вимагає $O(f(n)^2)$ часу для машини M (друга стрічка включає $O(f(n))$ пар, кожна з яких довжини $O(f(n))$ згідно з доведеним вище фактом, а сам пошук може бути виконаний за лінійний час). Обчислення результату складається з простих арифметичних операцій (*ADD*, *SUB*, *HALF*) на множині цілих чисел довжини $O(f(n))$, які можна виконати в часі $O(f(n))$. ■

4.2.11. Паралельні машини вільного доступу (PRAM)

Розглянемо узагальнення *RAM* машин на випадок паралельних обчислень — паралельна версія рандомізованої машини (Parallel Random Access Mashine (PRAM)). Введемо поняття *CRCW PRAM*, де скорочення *CRCW* означає *PRAM* машину з одночасним записом у регістри і одночасним читанням з регістрів.

Нехай $\Pi = (\pi_1, ..., \pi_n)$ — програма деякої *RAM* машини, тобто Π є скінченною послідовністю команд *RAM* машини, яка складається з команд типу *READ*, *ADD*, *LOAD*, *JUMP* і т. д., що використовують вміст регістрів (чарунок пам'яті). Нехай, як і раніше, у нульовому регістрі, що називається акумулятором *RAM* машини, записаний результат роботи машини, тобто результат останньої виконаної команди. На кожному кроці обчислень *RAM* машина виконує команду, яка визначається значенням лічильника команд k, читаючи з регістрів і записуючи в регістри значення у відповідності з командами *RAM*. Нехай $I = (i_1, ..., i_m)$ — множина вхідних регістрів *RAM* машини.

Визначення 91. **Програмою** *PRAM є послідовність* $P = (\Pi_1, ..., \Pi_q)$ *програм RAM машин, по одній на кожну з q RAM машин. При цьому вважаємо, що кожна RAM машина виконує власну програму, має свій власний лічильник команд, власний акумулятор (акумулятором і-ї машини є і-й регістр), але всі регістри є спільними — усі RAM машини можуть з них читати і можуть писати в ці регістри. Припускається також, що кожна машина може читати з акумуляторів і писати в ці акумулятори, і в PRAM немає регістра з номером* 0.

Кількість машин q в PRAM має бути постійним, а функція $q(m, n)$ *залежить від числа елементів m у вхідних регістрах I, а також від сумарної довжини цих елементів* $n = l(I)$. *Текст програми також залежить від* $\boldsymbol{m}$ *і* $\boldsymbol{n}$. *Іншими словами, для кожного значення* $\boldsymbol{m}$ *і* $\boldsymbol{n}$ *маємо другу програму* $P_{m,n}$ *з іншим числом RAM машин* $q(m, n)$, *а це означає, що двохмірна сім'я* $P = (P_{m,n} : m, n \geq 0)$ *програм PRAM.*

Для того щоб запобігти виникненню певних труднощів, будемо розглядати одну сім'ю програм *PRAM* і вважати, що всі ці програми однорідні. Це означає, що існує ДМТ, яка для даного вхідного слова 1^m01^n генерує число процесорів $q(m, n)$ разом з програмами $P_{m,n} = (\Pi_{m,n,0}, \Pi_{m,n,1}, \ldots, \Pi_{m,n,q(m,n)})$, використовуючи логарифмічну пам'ять. Як правило, функція $q(m, n)$ залежить тільки від m, але допускається щоб число процесорів залежало також і від числа вхідних елементів і від їх сумарної довжини. Це необхідно, оскільки залежно від проблеми *PRAM* може знадобитися «великий паралелізм» для великих чисел, які можуть бути присутні в даних.

Визначення 92. *Конфігурацією PRAM називається кортеж* $(k_1, k_2, \ldots, k_{q_{m,n}}, R)$, *що складається з лічильників команд усіх машин RAM і з R — сукупності поточних значень регістрів (див. означення RAM). Розширення відношення переходів на таких конфігураціях очевидно: за один крок і-та RAM виконує команду, яка визначається значенням її лічильника команд* k_i, *читаючи вміст з відповідних регістрів або зі свого акумулятора, яким є регістр і. Оскільки дозволені одночасні запис у регістри і читання з регістрів, то приймається таке правило:*

одержане значення RAM завжди записує в регістр з найменшим номером!

Нехай $F : N^k \to N^k$ *— деяка функція і* $P = (P_{m,n} : m, n \geq 0)$ *— однорідна сім'я програм PRAM. Крім того, нехай f і g будуть функ-*

ціями із N в N. Будемо говорити, що P обчислює функцію F у паралельному часі f на g процесорах, якщо для кожної пари чисел $m, n \geq 0$ *машина* $P_{m,n}$ *має такі властивості:*

1) вона має $q(m, n) \leq g(n)$ *процесорів;*

2) якщо програма PRAM застосовна до вхідних даних $I = (i_1,..., i_m)$, *що складаються із m цілих чисел із сумарною довжиною бітів* $l(I)$, *то всі* $q(m, n)$ *машин RAM дійдуть до стану HALT не більш ніж через* $f(n)$ *кроків і в цей момент на* $k \leq q(m, n)$ *перших регістрах буде записано значення функції*

$$F(i_1,..., i_m) = (o_1,..., o_k).$$

Визначена таким чином *PRAM* певною мірою є ідеалізованою машиною, але дуже потужною моделлю для виконання паралельних обчислень. У зв'язку з цим розглядаються й інші варіації *PRAM* машин, які будуються за допомогою різних обмежень, що накладаються на операції запису і читання, а також на множини регістрів, які при цьому використовуються. Більш слабкою моделлю, ніж *CRCW PRAM*, є *CREW PRAM* машини (від англійського *Concurent Read, Exclusive Write*), в яких дозволяється одночасне читання вмісту регістрів, але в кожний момент часу тільки одному з процесорів дозволяється писати в регістр. Ресурсом пам'яті в цьому випадку є число регістрів у послідовності обчислень c_1, c_2,..., а пам'ять, яка при цьому використовується, дорівнює максимальному числу k, такому, що клітинка c_k є доступною певному процесору під час обчислень. Час визначається дещо складніше. Вважається, що всі процесори працюють синхронно — один крок обчислень в один момент часу. Тоді час для одного кроку обчислень є максимумом за всіма процесорами «напівлогарифмічної» величини вартості для операції, що виконується цим процесором. Напівлогарифмічна вартість деякої операції з розміщеним в i-му регістрі числом N, довжина запису якого дорівнює n, виражається величиною $(1 + (\log N + \log i) / \log n)$. У цій моделі третім ресурсом є число процесорів, що використовуються. Сім'я *RAM* машин — це просто *PRAM* з одним єдиним процесором.

Нарешті, третя модель *PRAM* машини — це модель *EREW*, в якій заборонено як одночасне читання, так і одночасний запис у регістри.

Описаними вище трьома моделями не вичерпується сімейство *PRAM* машин. Існує ще принаймні два види *CRCW PRAM* машин, які визначаються залежно від того, яким способом виконується запис у регістри. Розглянута вище модель *CRCW PRAM* — це так зва-

на *PRIORITY CRCW PRAM*, у якій пріоритет запису вибирається відповідно до номера процесора. Більш слабкою, зате більш реальною є модель *ARBITRARY CRCW PRAM*, у якій машина довільним чином вибирає значення, яке записується в регістр (усі інші програми мають бути готовими працювати в ситуації, що враховує всі можливі випадки такого запису). Нарешті, у моделі *COMMON CRCW PRAM* усі процесори, що бажають записати деяке значення до одного й того самого регістра, повинні записувати одне й те саме значення. Отже, існує п'ять моделей *PRAM* машин, які розміщені відповідно до зростання їх обчислювальної потужності:

EREW, *CREW*, *COMMON CRCW*, *ARBITRARY CRCW*, *PRIORITY CRCW*

4.2.12. Контрольні питання, задачі і вправи

1. Дайте означення детермінованої МТ, недетермінованої МТ і багатострічкової МТ.

2. Дайте означення класу складності в часі і в пам'яті.

3. Побудуйте детерміновану МТ і багатострічкову МТ для обчислення функції:

   a) $o(x_1,..., x_n) = 0$ (нуль функція);

   b) $I_m^n(x_1,..., x_n) = x_m$ (селекторна функція);

   c) $f(x, y) = x + y$ (операція додавання натуральних чисел).

4. Подайте скінченний граф $G = (V = \{1, 2, 3\}, E = \{(1, 2), (1, 3), (2, 3), (3, 1)\}$ як слово в алфавіті $X = \{0, 1\}$.

5. Як можна записати числа-аргументи 12 і 9 для операції додавання на стрічці МТ у двійковій системі числення?

6. Покажіть, що операція додавання двох натуральних чисел виконується в поліноміальному часі. Чи можна побудувати більш ефективний алгоритм додавання двох натуральних чисел?

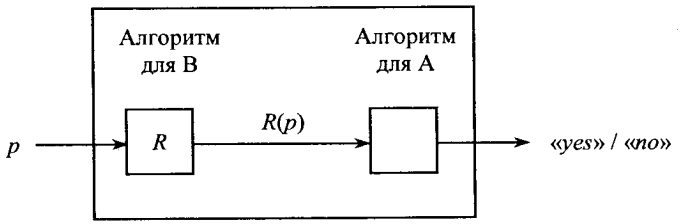
7. Дайте означення RAM-машини і побудуйте RAM-програми для обчислення значень функцій a)—c) з прикладу 3).

8. Дайте означення PRAM-машини. Побудуйте PRAM-програму для додавання двох натуральних чисел. Дайте оцінку часової складності одержаної PRAM-машини.

9. Представити слова $p = aba$ і $q = abacccdabaabc$, записані в алфавіті $X = \{a, b, c\}$, словами в алфавіті $\{0, 1\}$. Побудуйте ДМТ, яка перевіряє, є чи ні слово p підсловом слова q.

4.3.1. Редукція

Для того щоб можна було порівнювати між собою проблеми за складністю їх розв'язання, потрібне поняття, яке дозволяє точно окреслити ситуацію, коли одна проблема є такого ж ступеня складності, як і друга. Таким поняттям у теорії складності обчислень є поняття **редукції**. Неформально редукція проблеми B до проблеми A означає існування функції R, у результаті якої для довільного окремого випадку проблеми B є еквівалентний йому окремий випадок $R(p)$ проблеми A. Під еквівалентністю двох окремих випадків розуміють те, що відповідь на питання, чи є $R(p)$ позитивним прикладом проблеми A, буде одночасно відповіддю на питання, чи є p позитивним прикладом проблеми B. Іншими словами, розв'язати проблему B для прикладу p — це однаково, що розв'язати проблему A для прикладу $R(p)$.



Редукція B до A

Якщо має місце ситуація, що показана на вищенаведеному рисунку, то говорять, що проблема A є не менш важкою, ніж проблема B при одному застереженні. Це застереження стосується редукції R: редукція R не повинна бути занадто складною в обчислювальному плані. Якщо не зробити ніяких обмежень відносно цієї складності, то можна прийти до абсурдних наслідків. Тому під редукцією розуміють редукцію в логарифмічній пам'яті, формальне означення якої подано нижче.

Визначення 93. *Будемо говорити, що мова* L_1 **редукується до мови** L_2, *якщо існує словарна функція* R (*функція, визначена на словах і зі значеннями у множині слів*), *яка обчислюється за допомогою ДМТ в логарифмічній пам'яті* (*тобто* $O(\log n)$) *така, що для кожного слова* p *має місце*: $p \in L_1$ *тоді і тільки тоді, коли* $R(p) \in L_2$. *При цьому функція* R *називається* **редукцією** L_1 *до* L_2.

З цього означення безпосередньо випливає, що редукції є поліноміальними алгоритмами, і це підтверджується таким твердженням.

Теорема 87. *Якщо R є редукцією, яка обчислюється за допомогою деякої ДМТ M, то для кожного вхідного слова p машина M закінчує свою роботу через поліноміальне число кроків.*

Доведення. Існує тільки $O(n \cdot c^{\log n})$ можливих конфігурацій ДМТ M для даного слова p, де $n = l(p)$. Оскільки машина M детермінована, то жодна з її конфігурацій не може повторитися під час роботи M (повторення конфігурації означало б, що M ніколи не зупиняється). Отже, обчислення не можуть перевищувати довжини $O(n^k)$ для деякої константи k. Тепер залишається зауважити, що довжина $R(p)$ теж виражається поліномом, оскільки її значення обчислюється в поліноміальному часі (а на кожному кроці обчислень машина Тьюрінга може записати тільки один новий символ на стрічку). ■

Приклад 2.4.1. Покажемо, що проблема ГАМІЛЬТОНІВ ШЛЯХ редукується до проблеми ВИКОНУВАНІСТЬ. Формулювання цих проблем таке (необхідні поняття можна знайти в розділах 5, 7 і 9).

ВИКОНУВАНІСТЬ (ВИК)

ПРОБЛЕМА: Задано множину булевих змінних U і набір диз'юнктів C над множиною U.

ПИТАННЯ: Чи існує набір значень істинності, який виконує C?

ГАМІЛЬТОНІВ ШЛЯХ (ГШ) (див. підрозд. 5.4.7)

ПРОБЛЕМА: Дано граф $G = (V, E)$.

ПИТАННЯ: Чи правильно, що G включає гамільтонів шлях, тобто таку послідовність вершин $v_1, v_2, \ldots, v_n$ графа G, що $n = |V|$, $v_i \neq v_j$ при $i \neq j$ і $(v_i, v_{i+1}) \in E$ для всіх $1 \leq i \leq n - 1$.

Нехай $G = (V, E)$ — заданий граф. Скористаємося логічною формулою $R(G)$, яка виконується тоді і тільки тоді, коли в графі G існує гамільтонів шлях. Нехай G має n вершин (тобто $|V| = n$), тоді формула $R(G)$ має n^2 логічних змінних x_{ij} для $1 \leq i, j \leq n$. Неформально говорячи, x_{ij} являє собою той факт, що вершина j є i-ю вершиною на гамільтоновому шляху і, як легко пересвідчитися, x_{ij} може бути або істинною або хибною. Представимо формулу $R(G)$ в кон'юнктивній нормальній формі і опишемо її диз'юнкти (див. підрозділ 7.2). Ці диз'юнкти виражають вимоги до змінних x_{ij}, які, по суті, є вимогами до кодування гамільтонового шляху. По-перше, вершина j повинна появитися на цьому шляху. Ця вимога виконує диз'юнкт $(x_{1j} \vee x_{2j} \vee \ldots \vee x_{nj})$.

Побудуємо такі диз'юнкти для кожного j. Вершина j не може бути одночасно i-ю і k-ю вершиною шляху, що виражається диз'юнктом $(\neg x_{ij} \vee \neg x_{kj})$, який має бути повторений для всіх значень j і $i \neq k$. По-друге, має існувати вершина, яка буде на i-му місці шляху. Для цього додається диз'юнкт $(x_{i1} \vee x_{i2} \vee \ldots \vee x_{in})$, а також, враховуючи те, що ніякі дві вершини не можуть знаходитися на одному й тому самому місці, додаємо формулу $(\neg x_{ij} \vee \neg x_{ik})$ для всіх i і $j \neq k$. Нарешті, для кожної пари (i, j), яка не є ребром у графі G, j не може слідувати на гамільтоновому шляху безпосередньо за i, додаємо диз'юнкт $(\neg x_{ki} \vee \neg x_{k+1j})$. На цьому побудова формули $R(G)$ закінчується і $R(G)$ є кон'юнкцією всіх побудованих таким чином диз'юнктів.

Покажемо, що R є редукцією проблеми **ГШ** до проблеми **ВИК.** Для цього необхідно показати, що

1) для кожного графа G формула $R(G)$ виконується тоді і тільки тоді, коли граф G має гамільтонів шлях і

2) R можна обчислити в логарифмічній пам'яті.

Припустимо, що формула $R(G)$ виконана і послідовність T є послідовністю логічних значень, які виконують дану формулу. Оскільки T виконує всі диз'юнкти формули $R(G)$, то має бути правильним те, що для кожного j існує в точності одне i, таке, що $T(x_{ij}) = 1$, тому що в протилежному випадку диз'юнкти $(x_{1j} \vee x_{2j} \vee \ldots \vee x_{nj})$ і $(\neg x_{ij} \vee x_{kj})$ не можуть бути всі одночасно виконуваними. Аналогічно диз'юнкти $(x_{i1} \vee x_{i2} \vee \ldots \vee x_{in})$ і $(\neg x_{ij} \vee \neg x_{ik})$ гарантують, що для кожного i існує в точності одне j, таке що $T(x_{ij}) = 1$. Звідси отримуємо, що T дійсно являє собою перестановку $\pi(1),\ldots, \pi(n)$ вершин графа G, де $\pi(i) = j$ тоді і тільки тоді, коли $T(x_{ij}) = 1$. Крім того, диз'юнкт $(\neg x_{ki} \vee \neg x_{k+1j})$ для $(i, j) \notin E$ і $k = 1, 2,\ldots, n - 1$ гарантує те, що для кожного k пара $(\pi(k), \pi(k + 1)) \in E$. Але це означає, що послідовність $\pi(1), \pi(2),\ldots, \pi(n)$ є гамільтоновим шляхом у графі G.

З другого боку, припустимо, що граф G має гамільтонів шлях $\pi(1), \pi(2),\ldots, \pi(n)$, де π — відповідна перестановка. Тоді очевидно, що набір значень T, такий, що $T(x_{ij}) = 0$, де $\pi(i) \neq j$ виконує всі диз'юнкти формули $R(G)$.

Покажемо тепер, що редукція R може бути виконана в логарифмічній пам'яті $\log n$. Маючи заданий граф G, 3-стрічкова машина Тьюрінга M обчислює формулу $R(G)$ таким способом. Спочатку машина M записує бінарне число n — число вершин графа G, і виходячи з цього числа n, генерує на своїй вихідній стрічці один за другим диз'юнкти, які не залежать від графа (перші чотири групи диз'юнктів у формулі $R(G)$). Після цього машині потрібні три лічильники i, j і k, які використовуються для побудови індексів змінних у диз'юнктах. Для останньої групи диз'юнктів, які залежать від графа G, ма-

шина M знову генерує на своїй робочій стрічці один за другим диз'юнкти вигляду $(\neg x_{ki} \vee \neg x_{k+1j})$ для $k = 1, 2,..., n - 1$. Після завершення цієї генерації машина M перевіряє, використовуючи вхідне слово, є чи ні пара (i, j) ребром у графі G. Якщо це так, то виписує диз'юнкт на вихідній стрічці. Цим завершується побудова редукції проблеми **ГІП** до проблеми **ВИК**. ♠

Будемо позначати $L \leq_R L'$ той факт, що мова L редукується до мови L' за допомогою редукції R. Нехай маємо редукцію R_1 мови L_1 до мови L_2 і редукцію R_2 мови L_2 до мови L_3. Чи випливає звідси існування редукції R мови L_1 до мови L_3? Відповідь на це питання дає така теорема.

Теорема 88. *Якщо* $L_1 \leq_{R_1} L_2$ *і* $L_2 \leq_{R_2} L_3$, *то суперпозиція* $R_1 * R_2$ *редукцій* R_1 *і* R_2 *є редукцією мови* L_1 *до мови* L_3.

Доведення. З того, що R_1 і R_2 є редукціями мов, одразу випливає, що слово $p \in L$ тоді і тільки тоді, коли слово $R_2(R_1(p)) \in L_3$. Більш складною задачею є доведення того, що $R_1 * R_2$ можна обчислити в логарифмічній пам'яті $\log n$.

Найпростіший розв'язок міг би бути таким, як суперпозиція двох машин Тьюрінга, за якої вихідна стрічка однієї з машин є вхідною стрічкою другої машини. Нехай M_{R_1} і M_{R_2} — машини Тьюрінга, які обчислюють відповідно редукції R_1 і R_2 так, що спочатку обчислюється $R_1(p)$, а потім, використовуючи $R_1(p)$, обчислюється $R_2(R_1(p))$. На жаль, побудована таким чином машина повинна записати на своїй робочій стрічці слово $R_1(p)$, яке може виявитися набагато довшим, ніж $\log l(p)$.

Розв'язок цієї проблеми використовує такий метод. На робочій стрічці не будемо зберігати цілком значення проміжного слова, а замість цього будемо моделювати операції машини M_{R_2} на вхідному слові $R_1(p)$, запам'ятовуючи положення i її головки на вхідній стрічці (яка є вихідною для машини M_{R_1}). Положення i головки машини M_{R_2} зберігається у вигляді двійкового слова на новій стрічці машини M, значення якого спочатку дорівнює 1, тобто $i = 1$. Крім того, машина M, що обчислює редукцію $R_1 * R_2$, має також множину стрічок, які необхідні для моделювання решти обчислень машини M_{R_1} для $p \in L_1$.

Оскільки відомо, що спочатку головка машини M_{R_1} оглядає символ $\triangleright$, то легко змоделювати перший крок машини M_{R_2}. Коли головка на вхідній стрічці машини M_{R_1} пересувається вправо, то збіль-

шуємо i на одиницю і продовжуємо обчислення M_{R_1} на слові p (використовуючи окрему множину стрічок) стільки часу, скільки потрібно для обчислення $(i+1)$-го символу вхідного слова. Цим символом буде символ, який оглядається головкою машини M_{R_2}, і звідси продовжується подальше моделювання роботи машини M_{R_2}. Якщо головка залишається в тій самій позиції, то необхідно запам'ятати вхідний символ, який оглядається головкою. Але якщо головка на вхідній стрічці машини M_{R_2} переміщується вліво, то невідомо яким способом потрібно продовжувати моделювання, оскільки попередній символ, що записаний машиною M_{R_1}, уже запам'ятався. У цьому випадку діємо таким чином: зменшуємо i на одиницю і починаємо роботу машини M_{R_1} на слові p спочатку; на окремій стрічці обчислюємо записані в процесі моделювання символи вихідного слова і зупиняємося тоді, коли машина запам'ятає i-й символ. У цей момент можна поновити моделювання машини M_{R_2}. Легко зрозуміти, що така машина дійсно обчислює суперпозицію $R_1 * R_2$, використовуючи пам'ять $\log l(p)$ (нагадаємо, що довжина слова $R_1(p)$ виражається поліномом від $n = \log l(p)$ і, отже, i має не більш ніж $O(\log n)$ бітів). ■

Розглянемо одне узагальнення поняття редукції. Необхідність цього узагальнення полягає в тому, що поняття редукції можна застосувати до більш широкого класу проблем, ніж проблеми розв'язуваності. Цей більш широкий клас проблем складається із так званих «перебірних задач» або «задач пошуку».

Визначення 94. *Під* **перебірною задачею** П *розуміють множину* $D_П$ *скінченних об'єктів, які називаються* **індивідуальними задачами**, *причому для кожної індивідуальної задачі* $p \in D_П$ *відома множина* $S_П(p)$ *скінченних об'єктів, що називаються* **розв'язками** *задачі* p. *Будемо говорити, що алгоритм розв'язує перебірну задачу* П, *якщо одержавши на вході довільну індивідуальну задачу* $p \in D_П$, *цей алгоритм відповідає «по», якщо* $S_П(p)$ *пуста, і виробляє деякий розв'язок* $q \in S_П(p)$ *у протилежному випадку.*

З кожною перебірною задачею можна пов'язати деяке «словарне відношення».

Визначення 95. *Нехай* X *— скінченний алфавіт. Словниковим відношенням над алфавітом* X *називається бінарне відношення* $R \subseteq X^+ \times X^+$, *де* $X^+ = X^* \setminus \{e\}$ *— множина всіх непустих слів у алфавіті* X.

Нехай $L \subseteq X^+$ — деяка мова. Цю мову можна ототожнити зі словниковим відношенням

$$R = \{(p, q) : p \in X^+ \text{ і } q \in L\},$$

де p довільний фіксований символ із X. Зауважимо, що при цьому ігнорується питання належності пустого слова до мови, але це не впливає на питання обчислюваності, які тут розглядаються.

Визначення 96. *Вважають, що функція $f: X^* \to X^*$ реалізує словникове відношення R тоді і тільки тоді, коли для кожного $p \in X^+$ має місце $f(p) = e$, якщо не існує $q \in X^+$, такого, що $(p, q) \in R$; у протилежному випадку $f(x)$ дорівнює деякому слову $q \in X^+$, для якого $(p, q) \in R$. Словникове відношення R називається розв'язуваним, якщо існує ДМТ M, така що функція f_M, яка обчислюється машиною M, реалізує R.*

Відповідність між перебірними задачами і словниковими відношеннями виконується за допомогою схем кодування. Така схема для задачі П повинна видавати як слово, що є кодом кожної індивідуальної задачі $p \in D_П$, так і слово, що є кодом кожного розв'язку $q \in S_П(p)$. Нехай c — схема кодування перебірної задачі П, тоді задача П відповідає словниковому відношенню $R(П, c)$, яке означається таким чином:

$$R(П, c) = \{(p', q')\},$$

де $p' \in X^*$ (X — алфавіт для кодування) є кодом індивідуальної задачі $p \in D_П$, а $q' \in X^* \setminus \{e\}$ — код деякого розв'язку $q \in S_П(p)$ задачі p при схемі кодування c.

Визначення 97. *Вважають, що перебірна задача* П *при схемі кодування c розв'язувана поліноміальним алгоритмом, якщо існує ДМТ, яка розв'язує відношення $R(П, c)$ у поліноміальном часі.*

Необхідність узагальнення поняття поліноміальної редукції викликана тим, що довільна поліноміальна редукція проблеми П до проблеми П′ породжує алгоритм A для розв'язку проблеми П за допомогою гіпотетичної підпрограми розв'язку проблеми П′. Якщо дано перебірну задачу П і її індивідуальну задачу p, то цей алгоритм спочатку конструює індивідуальну задачу p' із П′, потім застосовує згадану підпрограму до p' і, нарешті, видає відповідь, яка виробляється цією підпрограмою, оскільки вона буде правильною відповіддю для p. Без урахування часу роботи підпрограми алгоритм A працює поліноміальний час. Таким чином, якщо підпрограма являє собою полі-

номіальний алгоритм розв'язку задачі П′, то вся описана процедура в цілому буде також поліноміальним алгоритмом розв'язання задачі П.

Зауважимо, що останнє твердження залишається справедливим, якщо алгоритм A буде використовувати підпрограму для П′ декілька (але не більше ніж поліноміальне число) разів, і навіть якщо та підпрограма буде служити для розв'язання перебірної задачі, а не для задачі розв'язання мови. Це є основою для запропонованого нижче узагальнення поняття редукції.

Визначення 98. *Говорять, що перебірна задача* П *зводиться за поліноміальний час до перебірної задачі* П′ *за Тьюрінгом, якщо існує алгоритм* A, *який розв'язує задачу* П *за допомогою гіпотетичної підпрограми* S *розв'язання задачі* П′, *такої, що коли* S *є поліноміальним алгоритмом для* П′, *то* A *є поліноміальним алгоритмом для* П.

Легко зрозуміти, що наведене на початку підрозділу означення редукції, є окремим випадком даного. Часто перше означення редукції називають трансформацією [54].

Зауважимо також, що існує таке поняття зведення, при якому ОМТ може тільки один раз робити запит до свого оракула і генерувати свою власну відповідь як відповідь, яка посилає оракул. Цей тип зведення називається в літературі **many — one** редукцією.

4.3.2. Повнота

З означення поняття редукції і теореми 88 випливає, що відношення редукованості є відношенням квазіпорядку, тобто воно є рефлексивним і транзитивним. Будем говорити, що дві проблеми однаково важкі, якщо існують редукції R і R', такі, що $П \leq_R П'$ і $П' \leq_{R'} П$. Неважко зрозуміти, що відношення «однаково важкі» є відношенням еквівалентності, а відношення редукованості буде частковим порядком на фактор-множині за цим відношенням еквівалентності. Надалі нас будуть цікавити *максимальні елементи* у цій фактор-множині. Для опису зазначених елементів використовується поняття повноти.

Визначення 99. *Нехай* C *— деякий клас складності і мова* L *є мовою із цього класу (тобто* $L \in C$*). Говорять, що мова* L *є* C*-повною, якщо довільна мова* $L' \in C$ *може бути редукована до мови* L.

Редукції можуть спростити доведення того, що деяка проблема належить даному класу складності. Але важливішим застосуванням

редукції є те, що за їх допомогою можна доводити, що деяка проблема не належить даному класу. Припустимо, що нам відомо, що проблема П не належить до класу C. Тоді, якщо можна знайти редукцію проблеми П до проблеми П′, то нам буде відомо, що проблема П′ теж не належить до класу C.

Найважчим питанням, звичайно, є пошук першої проблеми П, яка не належить до класу C. Редукція і тут стає корисною. Із теорем про ієрархію 81, 82, 83 випливає, що класи можуть включатися один до другого $C' \subseteq C$, тобто кожна проблема $П \in C'$ також належить класу C, але в класі C існує принаймні одна проблема П, яка не належить класу C.

Припустимо, що П — довільна проблема, до якої редукується деяка NP-повна проблема. Тоді незалежно від того, належить чи ні проблема П до класу NP, вона має ту властивість, що її неможливо розв'язати в поліноміальному часі, якщо $P \neq NP$. У цьому випадку говорять, що проблема П є NP-важкою. Ця обставина служить мотивацією для введення такого означення, яке є уточненням попереднього.

Визначення 100. *Нехай L — деяка мова і C — клас складності. Якщо $L' \leq_R L$ для всіх $L' \in C$, то говорять, що мова L є* **C-важкою (hard)**. *Якщо при цьому $L \in C$, то говорять, що мова L є* **C-повною (complete)**.

Визначення 101. *Клас складності C називається замкнутим відносно редукцій, якщо для довільної мови $L \in C$ із того, що L редукується до мови L' випливає, що мова L' теж належить до C, тобто із $L \in C$ і $L \leq_R L'$ випливає $L' \in C$.*

Має місце таке твердження.

Теорема 89. *Класи P, NP, $CoNP$, L, NL, $PSPACE$ і EXP замкнуті відносно редукцій [55].*

З цієї теореми випливає, що коли деяка P-повна проблема належить до класу C, то $P = C$, а якщо вона належить класу NP, то $P = NP$ і т.д. У цьому сенсі повні проблеми є основним засобом порівняння класів складності.

Теорема 90. *Якщо два класи складності C і C' замкнуті відносно редукцій і існує мова L, яка належить до обох класів, то $C = C'$.*

Доведення. Якщо мова L є спільним елементом для обох класів C і C', то для довільної мови L' із C' істинно $L' \leq_R L$. Отже, у силу замкнутості C' відносно редукцій, маємо $L \in C'$ і $C \subseteq C'$. Зворотне включення випливає із симетричності умови для мови.

Як було зазначено раніше, найбільш важким при доведенні того, що дана проблема належить деякому класу складності, є пошук першої задачі і доведення її належності до цього класу. Після того як відома одна задача із даного класу, процедура доведення належності до цього класу других проблем спрощується завдяки редукції. Для цього достатньо показати, що яка-небудь відома проблема з даного класу може бути редукована до даної проблеми. Таким чином, процес доведення зводиться до виконання таких кроків:

1) доведення того, що проблема П належить даному класу складності;
2) вибір відомої проблеми П′, що належить до цього класу складності;
3) виконання редукції П′ до П;
4) доведення того, що функція f виконує поліноміальне зведення.

4.4.1. P-повні проблеми

Як відомо, клас P непустий, оскільки до цього класу належать проблеми розпізнавання паліндромів і ізоморфізму графів. Покажемо належність до цього класу ще однієї важливої проблеми — проблеми ЗНАЧЕННЯ. Ця проблема формулюється так.

ЗНАЧЕННЯ

ПРОБЛЕМА: Дано логічну пропозиційну константну формулу (формула, в якій немає змінних).

ПИТАННЯ: Чи істинна дана формула на цих конкретних значеннях?

Алгоритм розв'язання проблеми ЗНАЧЕННЯ, очевидно, є поліноміальним, тому що обчислення виконуються без пошуку яких-небудь інтерпретацій. Але має місце і більш вагоме твердження.

Теорема 91. *Проблема ЗНАЧЕННЯ є P-повною.*

Доведення цієї теореми використовує поняття логічної функції і логічної мережі.

Визначення 102. *Функція* f: $\{true, false\}^n \to \{true, false\}$ *називається n-арною логічною функцією або функцією логічних значень, де true і false означають логічні константи, котрі означають відповідно значення «правда» і «фальш».*

Відомо, що довільна логічна функція може бути представлена формулою числення висловлювань і навпаки (див. підрозділ 7.1.2). Для представлення логічних функцій використовуються логічні мережі.

Визначення 103. *Логічною мережею, що відповідає логічній функції* $f(x_1,..., x_n)$, *називається ациклічний позначений орграф* $C = (V, E)$, *вершини якого* $V = \{1, 2,..., n\}$ *називаються* **воротами** *і позначені символами із множини* $Z = \{true, false, \neg, \vee, \wedge\} \cup \{x_1,..., x_n\}$, *де* x_i — *змінні функції* f. *Кожному з воріт* $i \in V$ *приписується позначка за допомогою функції позначок* $s(i)$ *таким чином. Якщо* $s(i) \in \{true, false, x_1,..., x_n\}$, *то вхідний степінь вершини* ***i*** *дорівнює* 0 (*тобто вершина* ***i*** *не має вхідних дуг*) *і вона називається* **вхідними воротами** *графа* C. *Якщо* $s(i) = \neg$, *то вершина* ***i*** *має вихідний степінь* 1, *а якщо* $s(i) \in \{\vee, \wedge\}$, *то вхідний степінь* ***i*** *дорівнює* 2. *Нарешті, ворота, які не мають вихідних дуг, називаються* **вихідними воротами** *або* **виходом мережі** C.

Часто розглядаються мережі, які мають декілька виходів, тобто одночасно обчислюють значення кількох логічних функцій. У такій мережі довільні ворота без вихідних дуг розглядаються як виходи.

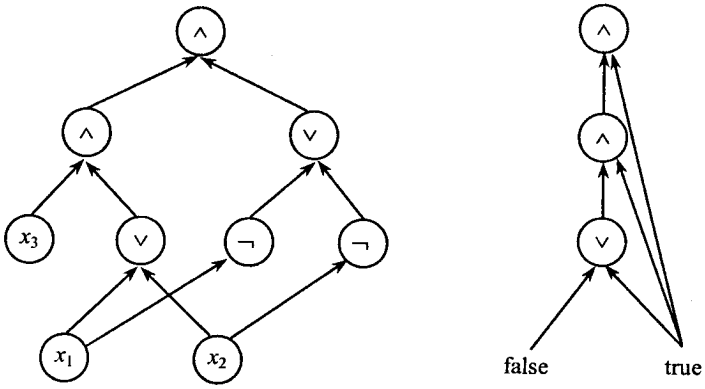
Семантика мережі означає логічне значення мережі для кожної інтерпретації змінних, які входять до цієї мережі. Індуктивне означення семантики таке. Нехай $X(C) = \{x \in X : s(i) = x$ для деяких воріт i в $C\}$, тоді якщо $T(i)$ означає значення воріт i, то

- якщо $s(i) = true(false)$, то $T(i) = true(false)$;
- якщо $s(i) \in \{x_1,..., x_n\}$, то $T(i) = T(s(i))$;
- якщо $s(i) = \neg$, то $T(i) = true(false)$, коли $T(j) = false$ ($true$) і $(j, i) \in E$ — ребро в C;
- якщо $s(i) = \vee$, то $T(i) = true$, коли хоча б одне із $T(j)$ або $T(k)$ дорівнює $true$ і $(j, i), (j, k) \in E$;
- якщо $s(i) = \wedge$, то $T(i) = true$, коли $T(j)$ і $T(k)$ дорівнюють $true$ і $(j, i), (j, k) \in E$.

Нарешті, **значенням** $T(C)$ **мережі** C є значення $T(n)$, де n — вихідні ворота мережі C.

Логічна мережа називається **константною,** якщо в ній немає жодної вершини, яка помічена символом змінної.

Нижче на рисунках показані логічні мережі для логічних функцій $(x_3 \wedge (x_1 \vee x_2)) \wedge (\neg x_1 \vee \neg x_2)$ і $(true \wedge (false \vee true)) \wedge true$:



Очевидно, що довільну n-арну логічну функцію можна представити і обчислити її значення за допомогою відповідної логічної мережі. Перейдемо тепер до доведення теореми.

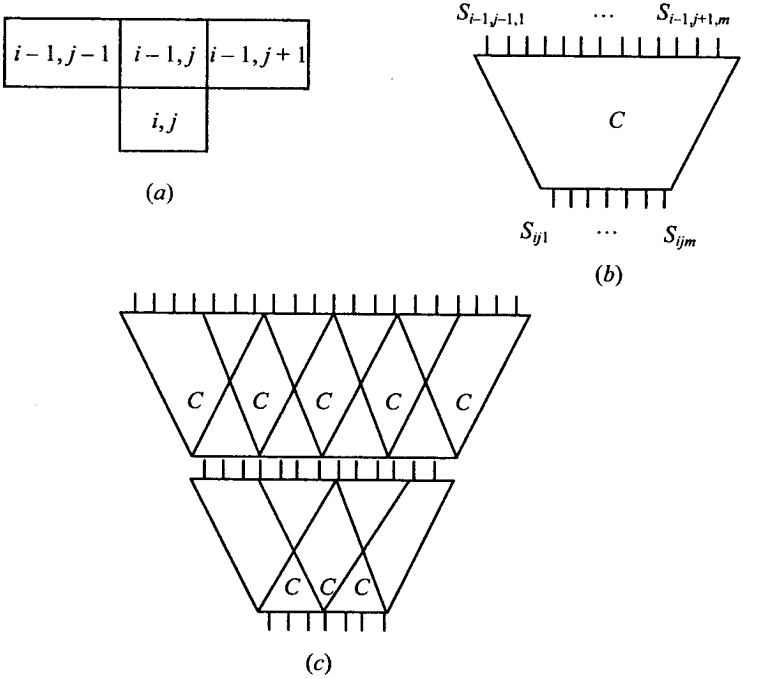
Доведення теореми 91. Як зазначалося раніше, проблема ЗНАЧЕННЯ розв'язується в поліноміальному часі. Покажемо, що для довільної мови $L \in P$ існує редукція R мови L до проблеми ЗНАЧЕННЯ.

Для даного вхідного слова p редукція $R(p)$ повинна бути мережею без змінних, такою що $p \in L$ тоді і тільки тоді, коли значення $R(p) = true$. Нехай M є машиною Тьюрінга, яка розв'язує мову L у часі n^k і нехай T буде таблицею обчислень машини M на слові p. Якщо $i = 0$ або $j = 0$, або $j = l(p)^k - 1$, то значення T_{ij} відомо (це буде j-й символ слова p або # в першому випадку, $\triangleright$ — у другому випадку і # — у третьому).

Розглянемо тепер довільний інший елемент T_{ij}. Значення T_{ij} включає значення j-ї позиції в i-й момент. Це значення залежить тільки від значення послідовності в j-й позиції або однієї з двох сусідніх позицій у момент $i - 1$. Іншими словами, T_{ij} залежить тільки від елементів $T_{i-1,j-1}$, $T_{i-1,j}$ і $T_{i-1,j+1}$ (див. нижче рис. (а)). Наприклад, якщо всі три елементи є символами із X, то тоді відомо, що в момент i головка машини не знаходиться ні на j-й позиції, ні на її найближчому сусідові і звідси отримуємо, що T_{ij} є таким самим, як і $T_{i-1,j}$. Якщо один з елементів $T_{i-1,j-1}$, $T_{i-1,j}$ або $T_{i-1,j+1}$ має вигляд σ_q, то T_{ij} може бути новим символом, що записується в момент i. Може трапитися, що цей символ $\sigma_{q'}$, якщо головка пересувається в позицію j, або може бути, що він дорівнює $T_{i-1,j}$. У кожному з випадків, для того щоб перевірити значення T_{ij}, необхідно перевірити тільки значення $T_{j-1,j-1}$, $T_{i-1,j}$ і $T_{i-1,j+1}$.

Нехай Γ означає множину всіх символів, які можуть з'явитися в таблиці (символи алфавіту машини Тьюрінга M, а також комбінації символів алфавіту і станів машини M). Закодуємо кожний символ $\sigma \in \Gamma$ вектором $(s_1,..., s_m)$, де $s_1,..., s_m \in \{0, 1\}$ і $m \le \log |\Gamma|$. Тепер таблицю значень можна розглядати як таблицю бінарних значень S_{ijl}, де $0 \le i \le n^k - 1, j \le n^k - 1$ і $1 \le l \le m$. Із спостережень попереднього абзацу стає зрозумілим, що кожне бінарне відношення S_{ijl} залежить тільки від $3m$ значень $S_{i-1,j-1,l'}$, $S_{i-1,j,l'}$ і $S_{i-1,j+1,l'}$, де l' набуває значення 1, 2,..., m. Це означає, що існує m логічних функцій $F_1,..., F_m$, кожна з яких має $3m$ аргументів, таких що для кожного $i, j \ge 0$ істинно

$$S_{ijl} = F_l\left(S_{i-1,j-1,1},...,S_{i-1,j-1,m},S_{i-1,j,1},...,S_{i-1,j+1,m}\right).$$



Конструкція мережі

Оскільки кожна логічна функція може бути представлена у вигляді логічної мережі, то для кожного $i = 1, 2,..., l(p)^k$ і $j = 1, 2,..., l(p)^k - 1$ існує мережа C з $3m$ входами і m виходами і яка обчислює двійковий код T_{ij} на основі двійкового коду $T_{i-1,j-1}$, $T_{i-1,j}$ і $T_{i-1,j+1}$. Мережа C

залежить тільки від машини M, має постійний розмір, що не залежить від довжини слова p (див. рис. (*b*)).

Тепер можна описати редукцію R мови L із **P**, яка розв'язується машиною M, до проблеми **ЗНАЧЕННЯ**. Для кожного заданого слова p редукція $R(p)$ складається із $(l(p)^k - 1)(l(p)^k - 1)$ копій мережі C (див. рис. (*c*)), по одній для кожного елемента T_{ij} таблиці обчислень. Позначимо через $C_{ij}(i,j)$-у копію C. Для $i \geq 1$ вхідні ворота мережі C_{ij} ототожнимо з вихідними воротами $C_{i-1,j-1}$, $C_{i-1,j}$ і $C_{i-1,j+1}$. Вхідними воротами всієї мережі є ворота, що відповідають першому рядку таблиці, а також першому і останньому стовпчику. Значення (*true*, *false*) відповідають відомим значенням цього рядка і стовпчиків. Нарешті, вхідні ворота $R(p)$ є першими вхідними воротами мережі $C_{l(p)^k-1,1}$ (без обмеження загальності, покладаємо тут, що машина M завжди закінчує обчислення в стані «*yes*» або «*no*», який записаний другим символом на стрічці, а також те, що кодування стану «*yes*» починається символом 1, а стану «*no*» — 0). Цим завершується опис мережі $R(p)$.

Покажемо, що значення мережі $R(p) = true$ тоді і тільки тоді, коли $p \in L$. Припустимо, що значення мережі $R(p) = true$. За допомогою математичної індукції за числом i можна показати, що значення вхідних воріт мережі C_{ij} дають двійкове представлення таблиці обчислень машини M на слові p. Оскільки значення мережі $R(p) = true$, то елемент $T_{l(p)^k-1,1}$ має значення «*yes*». Звідси випливає, що ця таблиця є сприймаючою, а це означає, що МТ M сприймає слово p і $p \in L$.

Навпаки, якщо $p \in L$, то таблиця обчислень є сприймаючою і звідси отримуємо, що значення мережі $R(p)$ дорівнює *true*, що й було потрібно.

Залишається показати, що редукцію R можна виконати в пам'яті $\log l(p)$. Оскільки мережа C визначена і залежить тільки від M, то обчислення R полягає в тому, щоб побудувати вхідні ворота (а для цього потрібно тільки перевірити значення символів слова p і знайти числа від 1 до $l(p)^k$) мережі C, а також з'єднання відповідних вхідних і вихідних воріт цих копій. Усі ці задачі потребують тільки безпосередніх операцій на індексах і тому їх легко виконати в пам'яті $O(\log l(p))$. ■

На завершення наведемо одну важливу **P**-важку проблему — проблему лінійного програмування.

ЛІНІЙНЕ ПРОГРАМУВАННЯ (ЛП)

ПРОБЛЕМА: Дано вектори, координати яких є цілими числами, $V = (v_i[1], v_i[2],..., v_i[n])$, $1 \le i \le m$ і $D = (d[1], d[2],..., d[m])$ і $C = (c[1], c[2],..., c[n])$ і ціле число B.

ПИТАННЯ: Чи існує вектор $X = (x[1], x[2],..., x[n])$ з раціональними координатами, такий, що $C \cdot X \ge B$ і $V_i \cdot X \le d[i]$ для всіх i, $1 \le i \le m$?

4.4.2. NP-повні проблеми

Розглянемо деякі з найбільш відомих NP-повних проблем. Почнемо з проблеми, якій випала честь бути першою проблемою в класі NP. Формулювання цієї проблеми наводилося вище і для неї було доведено, що вона є NP-повною проблемою. Цей результат тепер відомий як теорема Кука, за ім'ям автора, який уперше її довів.

Теорема 92. *Проблема ВИКОНУВАНІСТЬ є NP-повною проблемою.*

Доведення цієї важливої теореми досить складне і його можна знайти, наприклад, у [57, 55], а тут буде показана тільки редукція цієї проблеми до однієї з проблем, наведених нижче.

3-ВИКОНУВАНІСТЬ (3-ВИК)

ПРОБЛЕМА: Дано набір диз'юнктів $C = \{c_1, c_2,..., c_m\}$ на скінченній множині змінних U, таких, що $|c_i| = 3$, $1 \le i \le m$.

ПИТАННЯ: Чи існує на U набір значень істинності, при якому виконуються всі диз'юнкти із C?

ТРИМІРНЕ СПОЛУЧЕННЯ (3-С)

ПРОБЛЕМА: Дано множину $M \subseteq W \times X \times Y$, де W, X, Y — множини, що не перетинаються і мають однакову кількість елементів q.

ПИТАННЯ: Чи правильно, що M включає тримірне сполучення, тобто таку підмножину $M' \subseteq M$, що $|M'| = q$ і ніякі два різних елементи із M' не мають жодної однакової координати?

ВЕРШИННЕ ПОКРИТТЯ (ВП)

ПРОБЛЕМА: Дано граф $G = (V, E)$ і додатне ціле число K, $K \le |V|$.

ПИТАННЯ: Чи існує в графі G вершинне покриття, яке складається не більш ніж із k елементів, тобто така підмножина $V' \subseteq V$, що $|V'| \le k$ і для кожного ребра $(u, v) \in E$ принаймні одна із вершин u або v належить множині V'?

КЛІКА

ПРОБЛЕМА: Дано граф $G = (V, E)$ і додатне ціле число k, $k \le |V|$.

ПИТАННЯ: Чи правильно, що G включає деяку кліку потужності не меншої k, тобто таку підмножину $V' \subseteq V$, що $|V'| \ge k$ і довільні дві вершини із V' з'єднані ребром із E?

ГАМІЛЬТОНІВ ЦИКЛ (ГЦ)

ПРОБЛЕМА: Дано граф $G = (V, E)$.

ПИТАННЯ: Чи правильно, що G має гамільтонів цикл, тобто таку послідовність вершин $v_1, v_2,..., v_n$ графа G, що $n = |V|$, $(v_n, v_1) \in E$ і $(v_i, v_{i+1}) \in E$ для всіх $1 \le i \le n - 1$.

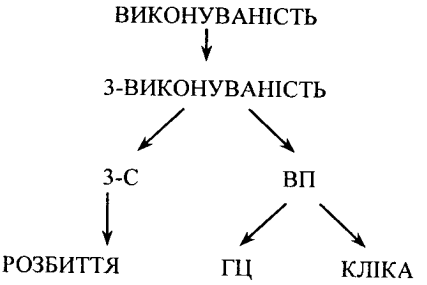
РОЗБИТТЯ

ПРОБЛЕМА: Дано скінченну множину A і цілі числа (вартості) $s(a) \in Z$ для кожного $a \in A$.

ПИТАННЯ: Чи існує підмножина $A' \subseteq A$ така, що

$$\sum_{a \in A'} s(a) = \sum_{a \in A \setminus A'} s(a)?$$

Доведення NP-повноти для зазначених вище проблем проводиться за такою схемою редукції:



Покажемо редукцію проблеми ВИКОНУВАНІСТЬ до проблеми 3-ВИК. Проблема 3-ВИК є просто обмеженим варіантом проблеми ВИКОНУВАНІСТЬ, у якому кожний диз'юнкт має рівно три літери. Завдяки своїй простій структурі ця проблема часто застосовується при доведенні NP-повноти.

Теорема 93. *Проблема 3-ВИК є NP-повною.*

Доведення. Зауважимо, що 3-ВИК $\in NP$. Це випливає з того, що недетермінованій МТ необхідно вгадати тільки набір значень істинності (інтерпретацію) змінних і за поліноміальний час перевірити, будуть чи ні істинними при такому наборі значень усі задані трьохлітерні диз'юнкти.

Редукуємо проблему ВИК до проблеми 3-ВИК. Нехай $U = \{u_1, u_2, \ldots, u_n\}$ — множина змінних і $C = \{c_1, c_2,..., c_m\}$ — довільний набір диз'юнктів, який визначає яку-небудь індивідуальну задачу з ВИК. Побудуємо набір C' трьохлітерних диз'юнктів на деякій множині змінних U', такий, що C' виконується тоді і тільки тоді, коли виконується C.

Набір C' будуватиметься шляхом заміни кожного окремого диз'юнкта $c_i \in C$ еквівалентним набором C'_i трьохлітерних диз'юнктів на множині U висхідних змінних і множині U'_i деяких додаткових змінних, причому змінні з U'_i будуть використовуватися тільки в диз'юнктах із C'_i. Іншими словами,

$$U' = U \cup \left(\bigcup_{j=1}^{m} U'_j \right)$$

і

$$C' = \bigcup_{j=1}^{m} C'_j.$$

Таким чином, потрібно тільки показати, як виходячи з c_i можна побудувати C'_j і U'_j.

Нехай c_i задається множиною $\{z_1, z_2, \ldots, z_k\}$, де z_i — літери з множини U. Спосіб побудови C'_j і U'_j залежить від значення k.

Випадок 1. $k = 1$. Тоді $U'_j = \{y_j^1, y_j^2\}$,

$$C'_j = \{\{z_1, y_j^1, y_j^2\}, \{z_1, y_j^1, \neg y_j^2\}, \{z_1, \neg y_j^1, y_j^2\}, \{z_1, \neg y_j^1, \neg y_j^2\}\}.$$

Випадок 2. $k = 2$. Тоді $U'_j = \{y_j^1\}$,

$$C'_j = \{\{z_1, z_2, y_j^1\}, \{z_1, z_2, \neg y_j^1\}\}.$$

Випадок 3. $k = 3$. Тоді $U'_j = \varnothing$, $C'_j = \{\{c_j\}\}$.

Випадок 4. $k > 3$. Тоді $U'_j = \{y_j^i : 1 \le i \le k-3\}$,

$$C'_j = \{\{z_1, z_2, y_j^1\}\} \cup \{\{\neg y_j^i, z_{i+2}, y_j^{i+1}\} : 1 \le i \le k-4\}\} \cup \{\{\neg y_j^{k-3}, z_{k-1}, z_k\}\}.$$

Для доведення того, що тут насправді має місце редукція, необхідно показати, що набір диз'юнктів C' виконується тоді і тільки тоді, коли виконується набір диз'юнктів C.

Припустимо спочатку, що $t : U \to \{true, false\}$ є інтерпретацією, в якій виконується C. Покажемо, що інтерпретація t може бути продовжена до інтерпретації $t' : U' \to \{true, false\}$, яка виконує C'. Оскільки змінні у множині $U' \setminus U$ діляться на групи U'_i і оскільки змінні в кожній групі U'_i входять тільки в диз'юнкти, які належать C'_i, то достатньо показати, що t може бути продовжена на кожну з множин U'_i окремо і в кожному випадку потрібно перевірити, що виконуються всі диз'юнкти із C'_i.

Це можна зробити таким чином. Якщо U'_i побудована, як у випадку 1 або 2, то диз'юнкти із C'_j вже виконані за допомогою інтер-

претації t, тому t' може бути продовжена на U'_i довільним чином, наприклад, якщо покласти $t'(y) = true$ для всіх $y \in U'_i$. Якщо U'_i побудована, як у випадку 3, то U'_i — пуста і єдина диз'юнкція в C'_i вже виконана за допомогою t. Залишається тільки випадок 4, який відповідає диз'юнкції $\{z_1,..., z_k\}$ із C, причому $k > 3$. Оскільки t — виконуюча інтерпретація для C, то знайдеться таке ціле число l, що z_l при інтерпретації t набуває значення *true*. Якщо $l = 1$ або 2, то покладаємо $t'(y'_i) = false$, $1 \le i \le k - 3$. Якщо $l = k - 1$ або k, то покладаємо $t'(y'_i) = false$, $1 \le i \le k - 3$. У протилежному випадку покладаємо $t'(y'_i) = true$, при $1 \le i \le l - 2$ і $t'(y'_i) = false$, при $-1 \le i \le k - 3$. Легко перевірити, що така інтерпретація забезпечує виконання всіх диз'юнктів із C'_i, тому інтерпретація t' виконує всі диз'юнкти з C'. Навпаки, якщо t' — деяка виконуюча інтерпретація для C', то легко перевірити, що обмеження t' на змінні з множини U є виконуючою інтерпретацією для C. Таким чином, C' виконується тоді і тільки тоді, коли виконується C.

Для того щоб перевірити, що ця редукція виконується в поліноміальному часі, достатньо зауважити, що число трьохлітерних диз'юнктів у C' обмежене поліномом від mn. Отже, розмір індивідуальної задачі 3-ВИК обмежений зверху деяким поліномом від розміру відповідної задачі ВИК, а оскільки всі деталі побудови редукції очевидні, то неважко перевірити, що ця редукція є поліноміальною.■

4.5. Класи складності DP, FP, FNP і поліноміальна ієрархія

4.5.1. Класи DP, P^{NP}, FP, FNP

Існують проблеми, які неможливо віднести ні до класу P, ні до класу NP. Ці проблеми вимагають розширення даних класів. Одним з таких класів є клас DP.

Визначення 104. *Мова L належить до класу DP тоді і тільки тоді, коли існує дві мови $L_1 \in NP$ і $L_2 \in CoNP$ такі, що $L = L_1 \cap L_2$.*

Зауважимо, що клас DP не збігається з класом $NP \cap CoNP$. Між цими класами мов існує величезна різниця, яка полягає хоча б у тому, що перетин $NP \cap CoNP$ стосується класів мов, а не самих мов, як в означенні класу DP. Друга важлива різниця між $NP \cap CoNP$ і DP полягає в тому, що клас DP суто синтаксичний і тому включає повні проблеми. Такою проблемою є така:

ВИКОНУВАНІСТЬ-НЕВИКОНУВАНІСТЬ (ВИК-НЕВИК)

ПРОБЛЕМА: Дано дві пропозиціональні формули ϕ і ϕ' у кон'юнктивній нормальній формі з трьома літерами в кожнім диз'юнкті.

ПИТАННЯ: Чи існує інтерпретація, в якій формула ϕ виконується, а формула ϕ' — ні?

Теорема 94. *Проблема ВИК-НЕВИК є DP-повною.*

Доведення. Для доведення необхідно показати, що існує дві мови $L_1 \in NP$ і $L_2 \in CoNP$, такі, що множина всіх позитивних прикладів ВИК-НЕВИК належить мові $L_1 \cap L_2$. Такі мови будуються просто: $L_1 = \{(\phi, \phi') : \phi$ виконувана$\}$ і $L_2 = \{(\phi, \phi') : \phi'$ невиконувана$\}$.

Покажемо тепер повноту проблеми. Нехай L — довільна мова з класу DP. Необхідно показати, що L редукується до проблеми ВИК-НЕВИК. Про мову L відомо тільки те, що існує дві мови $L_1 \in NP$ і $L_2 \in CoNP$, такі, що $L = L_1 \cap L_2$. В силу того, що проблема ВИК є NP-повною, то існує редукція R_1 мови L_1 до ВИК і редукція R_2 доповнення мови L_2 до ВИК. Тоді шукана редукція R мови L до проблеми ВИК має вигляд:

$$R(p) = (R_1(p), R_2(p))$$

для кожного заданого слова p. Отже, $R(p)$ є позитивним прикладом проблеми ВИК-НЕВИК тоді і тільки тоді, коли формула $R_1(p)$ виконується, а $R_2(p)$ — не виконується. А це має місце тоді і тільки тоді, коли $p \in L_1$ і $p \in L_2$ або, що те ж саме, що $p \in L$. ■

До класу DP, крім проблеми ВИК-НЕВИК, належить цілий ряд інших важливих проблем. Прикладами таких проблем є такі:

КРИТИЧНА-ВИКОНУВАНІСТЬ (К-ВИК)

ПРОБЛЕМА: Дано пропозиціональну формулу ϕ.

ПИТАННЯ: Чи є дана формула невиконуваною, але вилучення з неї довільного диз'юнкта робить її виконуваною?

ОДНОЗНАЧНА-ВИКОНУВАНІСТЬ (О-ВИК)

ПРОБЛЕМА: Дано пропозиціональну формулу ϕ.

ПИТАННЯ: Чи має дана формула єдину інтерпретацію, в якій вона виконується?

КРИТИЧНИЙ-ГАМІЛЬТОНІВ-ШЛЯХ (К-ГІП)

ПРОБЛЕМА: Дано граф G.

ПИТАННЯ: Якщо G не має гамільтонового шляху, то чи приводить до його появи додавання в граф G одного ребра?

На мови з класу DP можна дивитися як на клас мов, які розв'язуються за допомогою ОМТ. Така ОМТ задає оракулу два запитання і сприймає слово p тоді і тільки тоді, коли перша відповідь є «*yes*», а друга — «*no*». Таку ОМТ можна узагальнити на випадок довільної

заданої формули, яка генерує послідовність відповідей, що приводить до сприймання слова. Але більш цікавим узагальненням є дозвіл на довільну поліноміальну кількість запитань і дозвіл задавати питання, відповіді на які залежать від відповідей на попередні питання. Таким чином, одержуємо клас складності P як клас усіх мов, що розв'язуються за допомогою поліноміальної машини Тьюрінга з оракулом **ВИК**. Оскільки проблема **ВИК** є NP-повною, то на її місце можна підставити довільну іншу проблему з класу NP. Одержаний таким чином клас позначається P^{NP}.

Визначивши таким способом клас P^{NP}, можемо тепер визначити класи функцій *FP* і *FNP*. Необхідність введення цих класів полягає в тому, що крім проблем розв'язування мов, які вимагають лише відповідей «*yes*» або «*no*», існують проблеми обчислювального характеру, тобто проблеми обчислення значень функцій. Проблеми обчислення функцій називаються ще функціональними проблемами. Завдяки залежності між проблемами розв'язування і функціональними проблемами останні можуть бути теж формалізовані. Розглянемо означення класів функціональних проблем.

Визначення 105. *Нехай $R \subseteq X^* \times X^*$ — деяке бінарне відношення на множині слів у алфавіті X. Відношення R називається* **поліноміально розв'язуваним**, *якщо існує ДМТ, яка розв'язує в поліноміальному часі мову $L = \{p;\ q : (p, q) \in R\}$. Говорять, що* **відношення R є поліноміально збалансованим**, *якщо з того, що $(p, q) \in R$ випливає $l(q) \le l(p)^k$ для деякого $k \ge 1$, тобто довжина другої компоненти завжди поліноміально обмежена деяким поліномом від довжини першої компоненти.*

Теорема 95. *Нехай $L \subseteq X^*$ — деяка мова. $L \in NP$ тоді і тільки тоді, коли існує поліноміально розв'язуване і поліноміально збалансоване відношення R, таке, що $L = \{p : (p, q) \in R$ для деякого $q\}$.*

Доведення. Нехай таке відношення R існує. Тоді мова L розв'язується за допомогою такої НДМТ M. Машина M вгадує слово q довжини не більш $l(p)^k$, а потім використовує поліноміальний алгоритм для даних p; q з метою визначення належності $(p, q) \in R$. Якщо $(p, q) \in R$, то машина M сприймає вхідне слово, у протилежному випадку — не сприймає. Звідси відразу отримуємо, що обчислення, яке приводить до сприйняття слова p, існує тоді і тільки тоді, коли $p \in L$.

Для доведення теореми у зворотний бік, припустимо, що $L \in NP$, тобто що існує НДМТ M, яка розв'язує L у часі $l(p)^k$ для деякого k.

Визначимо відношення R таким чином: $(p, q) \in R \Leftrightarrow q$ є кодом сприймаючого обчислення машини M на слові p. Очевидно, що R є поліноміально збалансованим (оскільки МТ M є поліноміально обмеженою) і поліноміально розв'язуваним (оскільки можна з'ясувати за лінійний час, чи дійсно слово q є кодом сприймаючого обчислення машини M на слові p). Тепер із припущення, що МТ M розв'язує мову L, випливає рівність $L = \{p : (p, q) \in R$ для деякого $q\}$. ■

З цієї теореми випливає, що існує поліноміально роз'язуване і поліноміально збалансоване відношення R_L, таке, що для кожного слова p існує слово q, для якого $R_L(p, q)$ істинно тоді і тільки тоді, коли $p \in L$.

Визначення 106. *Функціональною проблемою, що асоціюється з мовою L (позначення FL) називається така проблема: для даного слова $p \in L$ знайти всі слова $q \in L$, такі, що істинно $R_L(p, q)$. Якщо таких слів не існує, то відповісти «по». Клас усіх функціональних проблем, що асоціюються з мовами з класу NP, позначається FNP. Клас проблем FP, що є підкласом FNP, складається із проблем, які розв'язуються в поліноміальному часі.*

Класи FP і FNP замкнуті відносно редукції. Основний зв'язок між цими класами і класами P і NP виражається таким твердженням.

Теорема 96. *$FP = FNP$ тоді і тільки тоді, коли $P = NP$* [55].

Неважко показати, що проблема ГВИК є FNP-повною. Клас проблем FP^{NP} є класом усіх словникових функцій із X^* в X^*, які можуть бути обчислені за допомогою поліноміальної МТ з оракулом ВИК. Насправді клас FP^{NP} є більш цікавим, ніж клас P^{NP}, оскільки він включає багато природних повних проблем. До таких FP^{NP}-повних проблем належить, наприклад, така:

МАХ-ВАРТ-ВИК

ПРОБЛЕМА: Дано множину диз'юнктів, кожному з яких приписано ціле число — вартість.

ПИТАННЯ: Знайти інтерпретацію, яка виконує дану множину диз'юнктів і має найбільшу сумарну вартість.

Має місце таке твердження.

Теорема 97. *Проблема МАХ-ВАРТ-ВИК є FP^{NP}-повною* [55].

4.5.2. Класи складності #P і ⊕P

На завершення даного підрозділу розглянемо ще два важливих класи складності $\#P$ и $\oplus P$. Почнемо з означення.

Визначення 107. *Нехай $Q \subseteq X^* \times X^*$ — деяке поліноміально збалансоване і поліноміально розв'язуване відношення. Проблемою обчислення, що пов'язана з відношенням Q, називається така проблема: скільки існує для заданого слова p таких слів q, що $(p, q) \in Q$? При цьому вимагається, щоб отриманий результат зображувався числом у двійковій системі числення.*
Клас #P складається з усіх проблем обчислення, які пов'язані з поліноміально збалансованими і поліноміально розв'язуваними відношеннями.

Якщо, наприклад, відношення Q є відношенням «q виконує формулу p», то відповідною проблемою обчислення буде проблема #ВИК. Якщо відношення Q є відношенням «q є гамільтоновим шляхом у графі p», то отримуємо проблему #ГШ і т.д. Ці проблеми є важливими проблемами із класу $\#P$. Проте має місце більш сильний результат.

Теорема 98. *Проблеми #ВИК і #ГШ є #P-повними* [54, 55].

Визначення 108. *Клас $\oplus P$ включає всі мови L, для яких існує НДМТ M, така, що для кожного вхідного слова p має місце $p \in L$ тоді і тільки тоді, коли число сприймаючих обчислень машини M на слові p є парним.*
Клас $\oplus P$ можна визначити і еквівалентним способом: $L \in \oplus P$, якщо існує поліноміально збалансоване і поліноміально обчислюване відношення R, таке, що $p \in L$ тоді і тільки тоді, коли число значень q, таких, що $(p, q) \in R$, є парним.

Для цього класу мають місце такі твердження.

Теорема 99. *Проблеми $\oplus$ВИК і $\oplus$ГШ є $\oplus P$-повними* [54, 55].

Теорема 100. *Клас $\oplus P$ замкнутий відносно доповнень* [54, 55].

4.5.3. Поліноміальна ієрархія

Після того, як визначені класи P^{NP} і NP^{NP}, можна визначити поліноміальну ієрархію класів складності. Існує думка про те, що клас NP^{NP} не є замкнутим відносно доповнень і, отже, необхідно розглядати клас $CoNP^{NP}$. Такого типу класи можна повторювати і далі. На цій ідеї і будується послідовність класів, що означаються нижче.

Визначення 109. *Поліноміальною ієрархією називається послідовність класів, які означаються індуктивно таким чином*:

$\Delta_0 P = \Sigma_0 P = \Pi_0 P = P$;

$\Delta_{i+1} P = P^{\Sigma_i P}$,

$\Sigma_{i+1} P = NP^{\Sigma_i P}$,

$\Pi_{i+1} P = CoNP^{\Sigma_i P}$ для всіх $i \geq 0$.

Сумарною поліноміальною ієрархією *або просто* **поліноміальною ієрархією** *називається клас*

$$PH = \bigcup_{i \geq 0} \Sigma_i P.$$

З цього означення отримуємо перший рівень ієрархії

$$\Delta_1 P = P,\ \Sigma_1 P = NP,\ \Pi_1 P = CoNP,$$

другий рівень ієрархії

$$\Delta_2 P = P^{NP},\ \Sigma_2 P = NP^{NP},\ \Pi_2 P = CoNP^{NP}$$

і т. д.

Безпосереднім узагальненням теореми 95 є така теорема.

Теорема 101. *Нехай $L \subseteq X^*$ — деяка мова й $i \geq 1$. $L \in \Sigma_i P$ тоді і тільки тоді, коли існує поліноміально збалансоване відношення R, таке, що $L = \{p : \text{існує таке } q, \text{ що } (p, q) \in R\}$, а за цих умов мова $L' = \{p; q: (p, q) \in R\}$ належить класу $\Pi_{i-1} P$.*

Доведення ведеться індукцією за числом i. При $i = 1$ справедливість твердження випливає безпосередньо з теореми 95. Припустимо, що $i > 1$ і відношення R існує. Необхідно показати, що $L \in \Sigma_i P$, а також описати недетерміновану поліноміальну ОМТ, оракулом якої є $\Sigma_i P$, і яка розв'язує мову L. Така ОМТ будується просто: ОМТ для заданого слова p знаходить відповідне слово q і запитує оракула $\Sigma_i P$, чи правильно, що $(p, q) \in R$ (точніше, так як R є $\Pi_{i-1} P$, то питає, чи вірно, що $(p, q) \in R$).

Навпаки, нехай $L \in \Sigma_i P$. Потрібно показати, що відношення R існує. Єдине, що нам відомо, так це те, що мова L може бути розв'язана за допомогою недетермінованої поліноміальної ОМТ M, котра використовує як оракул мову $K \in \Sigma_i P$. Оскільки $K \in \Sigma_i P$, то з припущення індукції випливає існування відношення S, яке розпізнається в часі $\Pi_{i-1} P$ і такого, що $z \in K$ тоді і тільки тоді, коли існує w, таке, що $(z, w) \in S$. Необхідно знайти поліноміально збалансоване і поліноміально розв'язуване відношення R для мови L, тобто допуск для кожного слова $p \in L$. Відомо, що $p \in L$ тоді і тільки тоді,

коли існує правильне обчислення для машини M^K на слові p, що сприймає це слово. Існуючим допуском для слова p буде слово q, яке є записом такого обчислення машини M^K (див. доведення теореми 95). Тепер, згадуючи те, що в даному випадку M^K є ОМТ, оракулом якої є $K \in \Sigma_i P$ і в якій деякі кроки обчислень представляють запитання до K. На одні з цих запитань буде відповідь «*yes*», а на інші — «*no*». Для кожного запитання з відповіддю «*yes*» допуск буде включати слово w_i — допуск для z_i, таке, що $(z_i, w_i) \in S$. А це означає, що відношення R можна означити таким чином: $(p, q) \in R$ тоді і тільки тоді, коли q є записом сприймаючого обчислення машини M на слові p разом з допуском w_i для кожного питання z_i, відповіддю на який при обчисленні q є відповідь «*yes*».

Покажемо, що перевірити належність $(p, q) \in R$ можна в часі $\Pi_{i-1}P$. Спочатку необхідно показати, що всі кроки обчислень машини M виконуються у відповідності до функції переходів M і що це можна перевірити в поліноміальному детермінованому часі. Далі для поліноміального числа пар (z_i, w_i) потрібно перевірити, чи правильно, що $(z_i, w_i) \in S$. Але це можна перевірити в часі $\Pi_{i-2}P$, а отже, і в часі $\Pi_{i-1}P$. Нарешті, для всіх запитань z_i', на які отримано негативну відповідь у процесі обчислень, потрібно перевірити, що дійсно $z_i' \in K$. Але оскільки $K \in \Sigma_i P$, то це чергове запитання з $\Pi_{i-1}P$. Остаточно отримуємо, що $(p, q) \in R$ тоді і тільки тоді, коли відповідь на всі вищезгадані запитання з класу $\Pi_{i-1}P$ отримають відповідь «*yes*». Зрозуміло, що таку перевірку можна виконати протягом одного обчислення $\Pi_{i-1}P$. ■

Подвійним до даного твердження є таке.

Теорема 102. *Нехай $L \subseteq X^*$ — деяка мова й $i \geq 1$. $L \in \Pi_i P$ тоді і тільки тоді, коли існує поліноміально збалансоване відношення R, таке, що $L = \{p : \text{для кожного } q, \text{такого, що } l(q) < l(p)^k, (p, q) \in R\}$, а за цих умов мова $L' = \{p; q : (p, q) \in R\}$ належить до класу $\Sigma_{i-1}P$.*

Доведення. Для доведення зауважимо, що $\Pi_i P$ — це в точності $Co\Sigma_i P$. ■

Використовуючи попередню теорему, можна сформулювати умови стабілізації поліноміальної ієрархії.

Теорема 103. *Якщо для деякого $i \geq 1$ має місце $\Sigma_i P = \Pi_i P$, то для кожного $j > i$ правильно*

$$\Sigma_j P = \Pi_j P = \Delta_j P = \Sigma_i P.$$

Доведення. Достатньо показати, що з рівності $\Sigma_i P = \Pi_i P$ випливає рівність $\Sigma_{i+1}P = \Sigma_i P$. Для цього розглянемо мову $L \in \Sigma_{i+1}P$. З теореми

101 випливає, що існує відношення R із $\Pi_i P$, для якого $L = \{p$: існує таке q, що $(p, q) \in R\}$. Але оскільки $\Sigma_i P = \Pi_i P$, то R належить до класу $\Sigma_i P$. А це означає, що $(p, q) \in R$ тоді і тільки тоді, коли існує $(p, q, z) \in S$ для деякого відношення $S \in \Pi_{i-1} P$, а також що $p \in L$ тоді і тільки тоді, коли існує таке слово q; z, що $(p, q, z) \in S$, де $S \in \Pi_{i-1} P$. А це означає, що $L \in \Sigma_i P$. ■

З цієї теореми отримуємо такий наслідок.

Наслідок 24. *Якщо $P = NP$ або $NP = CoNP$, то поліноміальна ієрархія стабілізується на першому рівні.*

Поліноміальна ієрархія становить інтерес з декількох точок зору. По-перше, вона відповідає важливій ієрархії «все більше й більше нерозв'язуваних проблем», яка називається **арифметичною ієрархією Кліні**. По-друге, різні рівні цієї ієрархії охоплюють певну кількість (хоча й не невелику) цікавих природних проблем; деякі з них є повними. Розглянемо одну з таких проблем.

МІНІМАЛЬНА БУЛЕВА МЕРЕЖА (МБМ)

ПРОБЛЕМА: Дано пропозиційну формулу.

ПИТАННЯ: Чи існує пропозиційна формула, що еквівалентна даній і яка має мінімальне число змінних?

Доведено, що **МБМ** належить до класу $\Pi_2 P$ і поки що не вдалося показати, що дана проблема належить до якого-небудь більш низького класу ієрархії. Відкритим також залишається питання про те, чи є проблема МБМ $\Pi_2 P$-повною.

Наступна сім'я проблем показує, що класи поліноміальної ієрархії непусті. Проблеми, які складають цю сім'ю, мають назву Q_iВНК — виконуваність формул логіки предикатів першого порядку, що мають i кванторів. Точне формулювання цих проблем таке.

Q_iВИКОНУВАНІСТЬ (Q_iВИК)

ПРОБЛЕМА: Дано формулу Ф логіки предикатів першого порядку з множиною змінних, розбитих на i груп $X_1, X_2, \ldots, X_i$.

ПИТАННЯ: Чи буде для кожної часткової інтерпретації змінних з групи X_1, існувати часткова інтерпретація змінних з X_2, така, що для кожної часткової інтерпретації змінних з групи X_3 і т. д. до X_i формула Ф виконується?

Розширюючи значення кванторів на групи змінних, проблему Q_iВИК можна записати так: чи є виконуваною формула

$$\forall X_1 \exists X_2 \forall X_3 \ldots Q X_i \Phi,$$

де $QX_j = \exists X_j$, якщо j парне, і $QX_j = \forall X_j$, якщо j непарне.

Мають місце такі твердження (див. [54, 55].

Теорема 104. *Для довільного $i \geq 1$ проблема Q_iВНК є $\Sigma_i P$-повною.*

Теорема 105. *Якщо існує PH-повна проблема, то поліноміальна ієрархія стабілізується на деякому скінченному рівні.*

Теорема 106. *$PH \subseteq PSPACE$.*

Зауважимо, що питання справедливості рівності PH і PSPACE залишається відкритим.

4.5.4. Клас складності NC

Розглянемо коротко один з важливих класів складності паралельних обчислень і його зв'язок з машинами *PRAM*. Означення цього класу використовує поняття логічної однорідної мережі.

Нехай C — деяка логічна мережа, тобто ациклічний орграф, вершини якого називаються воротами і розмічені символами алфавіту $\{true, false, \neg, \vee, \wedge\} \cup \{x_1, x_2, \ldots, x_n\}$. **Розміром** мережі C називається число воріт у ній, а її **глибиною** — число вершин найдовшого шляху в цій мережі.

Визначення 110. *Нехай $C = (C_1, C_2, \ldots)$, C_n — сім'я однорідних логічних мереж і $f, g : N \to N$ — функції. Говорять, що* **паралельний час** *сім'ї C вимагає часу $f(n)$, якщо для кожного n глибина мережі C_n не перевищує $f(n)$, а* **загальна робота** *сім'ї мережі C не перевищує $g(n)$, якщо для довільного $n \geq 0$ розмір мережі C_n не перевищує $g(n)$.*

Клас складності **PT/WK**$(f(n), g(n))$ означається як клас мов $L \subseteq X^*$ $(X = \{0, 1\})$, для яких існує сім'я однорідних мереж C, яка розв'язує мову L у паралельному часі $O(f(n))$ і використовує при цьому $O(g(n))$ одиниць роботи.

Визначення 111. *Клас складності* **NC** = **PT/WK**$(\log^k n, n^k)$ *включає всі проблеми, які розв'язуються в полілогарифмічному паралельному часі при загальній поліноміальній роботі.*

У термінах мов клас **NC**, як виявилося [55], у точності збігається з класом мов, що розв'язуються в полілогарифмічному паралельному часі за допомогою машин PRAM з поліноміальним числом процесорів. Таким чином, клас **NC** відповідає інтуїтивному поняттю «проблем, що розв'язуються за допомогою паралельних комп'ютерів»,

подібно до того, як клас **P** відповідає інтуїтивному поняттю класу «проблем, що розв'язуються на послідовних комп'ютерах».

Окремим випадком класу **NC** є клас $\mathbf{NC}_j$, який означається так:

$$\mathbf{NC}_j = \mathbf{PT/WK}(\log^j n, n^k)).$$

Це означає, що $\mathbf{NC}_j$ є підмножиною класу **NC**, де паралельний час не перевищує $O(\log^j n)$, а вільний параметр k означає, що допускається робота, яка може бути поліномом довільного степеня.

Очевидно, що $\mathbf{NC} \subseteq \mathbf{P}$. А питання рівності цих класів залишається відкритим, подібно до питання про рівність класів **P** і **NP**. Має місце таке твердження.

Теорема 107. *Якщо мова L редукується до мови $L' \in NC$, то $L \in NC$.*

Доведення. Нехай R — редукція мови L до мови $L' \in NC$. Неважко помітити, що існує машина Тьюрінга R', яка працює в логарифмічній пам'яті і яка сприймає (p, i) (де i — двійковий запис цілого числа не більшого за $|R(p)|$), тоді і тільки тоді, коли i-й біт $R(p)$ дорівнює одиниці. Розв'язуючи проблему досяжності в графі конфігурації R' для даних (p, i), можна обчислити i-й біт $R(p)$. Якщо всі такі проблеми розв'язуємо паралельно за допомогою мережі NC_2, то тим самим обчислюються всі біти $R(p)$. Коли вже маємо $R(p)$, то можемо використати мережу NC для мови L', щоб перевірити, що $p \in L$. Усе це можна зробити в NC. ■

Наслідок 25. *Якщо мова L редукується до мови $L' \in NC_j$ для $j \geq 2$, то $L \in NC_j$.*

4.5.5. Класи складності EXP, NEXP і вище

Продовжуючи побудову ієрархії класів складності, почнемо з експоненціальної пам'яті.

а) Наступним у класі ієрархії складності з пам'яті є клас

$$\mathbf{EXPSPACE} = \mathbf{SPACE}(2^{n^k}).$$

b) Наступним у класі ієрархії часової складності є клас, який називається **подвійною експонентою**, тобто

$$\mathbf{2\text{-}EXP} = \mathbf{TIME}(2^{2^{n^k}}).$$

c) Наступним у класі ієрархії часової складності є клас мов **2-NEXP** — клас мов, що розв'язуються НДМТ в часі $2^{2^{n^k}}$, тобто

$$\mathbf{2\text{-}NEXP} = \mathbf{NTIME}(2^{2^{n^k}}).$$

d) Клас часової складності **3-EXP** означається так само, як і попередній клас мов, але

$$\textbf{3-EXP} = \textbf{TIME}(2^{2^{2^{n^k}}})$$

і так далі.

e) Наступним у класі ієрархії часової складності є клас, що називається **елементарним**, і його означення має вигляд

$$\textbf{ELEMENTARY} = \bigcup_{t>1} \textbf{TIME}(2^{2^{\cdot^{\cdot^{2^{n^k}}}}}),$$

де t — кількість двійок в експоненті ($t = 2, 3, ...$).

f) Клас рекурсивних мов, який позначається R.

g) Клас рекурсивно перелічимих мов, який позначається RE, і клас мов $Co(RE)$, елементи якого є доповненнями рекурсивно перелічимих мов.

4.5.6. Арифметична складність алгоритмів

Як випливає з вищенаведеного, часова складність алгоритму виражається числом кроків машини Тьюрінга, необхідних для його реалізації, відносно довжини коду вхідних даних алгоритму. Цей підхід до вимірювання часової складності алгоритмів називають **складністю за Тьюрінгом** або **моделлю Тьюрінга**. Природним також є аналіз складності алгоритму в термінах числа арифметичних операцій, необхідних для його реалізації. Цей підхід до вимірювання часової складності алгоритмів називається **арифметичною складністю** або **арифметичною моделлю**. В арифметичній моделі довжина коду вхідних даних не враховується, а враховується лише їх число, у той час як у моделі Тьюрінга довжина коду вхідних даних є суттєвим параметром.

Часова складність алгоритму може бути поліноміальною в моделі Тьюрінга, але може не бути такою в арифметичній моделі і навпаки. Наприклад, алгоритм Евкліда для знаходження найбільшого спільного дільника двох цілих чисел є поліноміальним у моделі Тьюрінга, але не є таким в арифметичній моделі. Навпаки, поліноміальне число арифметичних операцій не може бути виконане на машині Тьюрінга, якщо довжина кодуючої послідовності вхідних даних не є поліноміально обмеженою.

Визначення 112. *Говорять, що алгоритм має* **строго поліноміальну оцінку часової складності**, *якщо він є поліноміальним за складністю в арифметичній моделі і поліноміальним за пам'яттю в моделі Тьюрінга.*

З цього означення випливає, що часова складність алгоритму залежить від того, яким чином зображуються (кодуються) раціональні числа і результати арифметичних операцій. Вважається, що коли a і b два цілих числа і відомо, що b ділить a без остачі, то раціональне число a/b зображується відповідним цілим числом. Якщо число a/b не є цілим, то вважається, що воно зображується в так званій **coprime** формі a/b, де НСД(a, b) = 1. Іншими словами, *coprime* форма — це зображення числа a/b у вигляді нескорочуваного дробу.

4.5.7. Контрольні питання, задачі і вправи

Контрольні питання

1. Дайте означення редукції, трансформації і повноти проблеми.
2. Яка проблема називається **P-**, **NP-**, **#P**-важкою?
3. Дайте означення класів складності **#P**, **⊕P**, **FP**, **FNP**, **NC**, **DP**.
4. Дайте означення поліноміальної ієрархії і наведіть класи цієї ієрархії, що відповідають рівням 0, 1, 2.
5. Які приклади проблем належать до класів ***#P***, **⊕P, FP, FNP, NC, DP**?

Задачі і вправи

1. Доведіть теорему 89.
2. Побудуйте редукцію проблеми **3-ВИК** до проблеми **ВП**.
3. Побудуйте редукцію проблеми **ВИК-НЕВИК** до проблеми **К-ВИК.**
4. Довести теореми 98—100.
5. Довести, що алгоритм Евкліда для знаходження НСД двох цілих чисел має поліноміальну часову складність у моделі Тьюрінга.
6. Знайти оцінку часової складності алгоритму Евкліда для знаходження НСД двох цілих чисел в арифметичній моделі.
7. Яка складність алгоритму множення двох натуральних чисел в арифметичній моделі?

5 ЕЛЕМЕНТИ ТЕОРІЇ ГРАФІВ

Виникнення теорії графів пов'язують з іменем Ейлера, який у 1736 р. не тільки розв'язав популярну на той час головоломку про кенігсберзькі мости [39, 13], а й знайшов критерій існування в графі спеціального маршруту (ейлерового циклу). Довгий час цей результат залишався єдиним результатом теорії графів, і лише в середині XIX століття переважно зусиллями Кірхгофа та Келі були одержані нові результати в теорії графів. Приблизно в цей самий час виникла знаменита проблема чотирьох фарб.

Хоча теорія графів виникла більше двох століть тому, проте її інтенсивний розвиток припадає лише на останні 50—60 років завдяки широкому застосуванню графів у теорії автоматів, теорії проектування, економіці, хімії, біології тощо. Оскільки 50—60 років для глибокої теорії — це відносно молодий вік, то термінологія в теорії графів ще й до нині змінюється. Виникають нові поняття і методи. У зв'язку з цим понятійний матеріал цього розділу може мати деякі розбіжності з іншими літературними джерелами щодо теорії графів, хоча в ньому зібрані більш менш сталі поняття, визначення і методи теорії графів.

5.1. Неорієнтовані графи, різновиди графів

5.1.1. Визначення графа

Нехай V — деяка непуста множина, $V^{(2)}$ — множина всіх неупорядкованих різних двохелементних підмножин множини V, а $MV^{(2)}$ — мультимножина множини $V^{(2)}$, тобто $MV^{(2)}$ може включати однакові пари елементів із V, причому цих пар може бути скільки завгодно. Як і раніше, декартовий квадрат множини V будемо позначати V^2.

Визначення 113. **Неорієнтованим мультиграфом** *G називається пара* (V, E), *де* $E \subseteq MV^{(2)}$. *Елементи множини V називаються* **вершинами**, *а елементи множини E* — **ребрами.** *Мультиграф* $G = (V, E)$ *називається просто* **неорієнтованим графом**, *якщо* $E \subseteq V^{(2)}$.

Підкреслимо, що довільний граф є мультиграфом, але не кожний мультиграф буде графом. Якщо $G = (V, E)$ — мультиграф, то E може мати декілька ребер виду (u, v). Такі ребра називаються **кратними ребрами.** Отже, граф є мультиграфом, у якого кратність кожного ребра дорівнює одиниці.

Мультиграф називається **скінченним**, якщо множини V і E обидві скінченні. Для скінченності графа, очевидно, достатньо лише скінченності множини його вершин V, оскільки скінченність V дає скінченність $V^{(2)}$, тобто граф називається **скінченним,** якщо скінченна множина його вершин. Скінченний граф з n вершинами називається **графом *n*-го порядку.**

Інколи розглядають графи, які мають ребра виду (u, u). Ребро такого виду називається *петлею*, а мультиграф, який має петлі, — **псевдографом**.

Надалі, коли не сказано супротивне, будемо розглядати тільки скінченні графи.

Говорять, що дві вершини u і v графа $G = (V, E)$ *суміжні*, якщо $(u, v) \in E$, і *несуміжні* в протилежному випадку. Якщо $(u, v) \in E$, то вершини u і v називаються *кінцями ребра* (u, v). У цьому випадку ще говорять, що ребро (u,v) з'єднує вершини u і v. Множину вершин графа, суміжних з деякою вершиною u, будемо позначати $См(u)$.

Із цих визначень випливає також, що різниця між графом і мультиграфом полягає в тому, що дві вершини в графі можуть бути з'єднані не більше ніж одним ребром, а в мультиграфі дві вершини можуть з'єднуватися більше ніж одним ребром.

Два ребра називаються *суміжними*, якщо вони мають спільний кінець. Зауважимо, що відношення суміжності як для вершин, так і для ребер графа є симетричним відношенням.

Вершина u і ребро e називаються *інцидентними*, якщо u є кінцем ребра e, і *неінцидентними* у протилежному випадку.

Степенем $n(u)$ вершини u графа називається число інцидентних їй ребер. Вершина степеня 0 називається **ізольованою**, а вершина степеня 1 — **висячою** або **кінцевою**. Ребро, інцидентне кінцевій вершині, також називається **кінцевим**. Наступне просте, майже очевидне, твердження дає зручну характеристику графів.

Лема 5 (*Лема про рукостискання*). *Сума степенів усіх вершин графа є парним числом.*

Дійсно, кожне ребро вносить до суми всіх степенів вершин графа число 2. Отже,

$$\sum_{v \in V} n(v) = 2|E|.$$

Якщо інтерпретувати кожне ребро як рукостискання двох людей, то при довільному числі рукостискань загальна кількість потиснутих рук буде парною. Звідси і назва леми.

Наслідок 26. *У довільному графі число вершин непарної степені парне.*

Дійсно, якби це було не так, то сума степенів усіх вершин графа не могла б бути парним числом, що суперечить лемі про рукостискання.

Графи зручно зображувати на площині або в просторі у вигляді діаграм, які складаються з точок і відрізків, що з'єднують деякі з цих точок. При цьому точки ототожнюються з вершинами графа, а відрізки — з його ребрами (див. рис. 5.1.1).

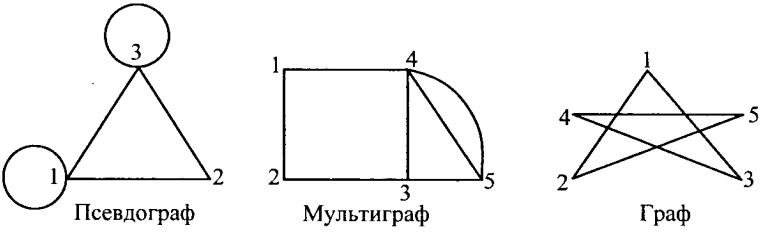


Рис. 5.1.1

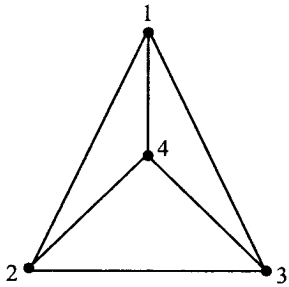
5.1.2. Різновиди графів

Розглянемо деякі різновиди графів, які часто зустрічаються.

1. Повні графи

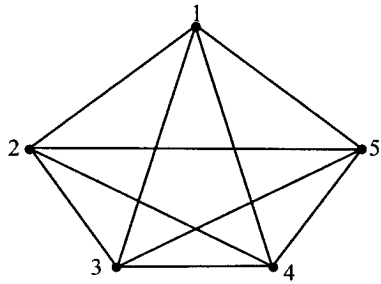
Визначення 114. *Граф, у якого дві довільні вершини суміжні, називається* **повним графом**.

Отже, якщо $G = (V, E)$ — повний граф, то $E = V^{(2)}$. Повний граф з n вершинами будемо позначати K_n. Графи K_4 і K_5 зображено на рис. 5.1.2 і 5.1.3.



Граф K_4

Рис. 5.1.2



Граф K_5

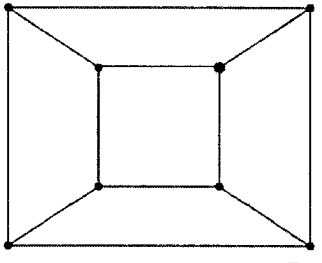
Рис. 5.1.3

2. Регулярні графи. Більш загальними, ніж повні, є регулярні графи.

Визначення 115. *Граф називається* **регулярним** *або* **однорідним**, *якщо всі його вершини мають один і той самий степінь. Якщо степінь кожної вершини дорівнює k, то граф називається* **регулярним графом степеня** *k.*

Отже, повний граф n-го порядку є регулярним графом степеня $n-1$. Регулярні графи степеня 3 називають також **кубічними** або **трьохвалентними** графами (ці графи викликають особливий інтерес у зв'язку із завданням розфарбування графів, яка буде розглянута пізніше). Відомим прикладом кубічного графа є граф Петерсена, який показано на рис. 5.1.4 (*б*).

а)



б)

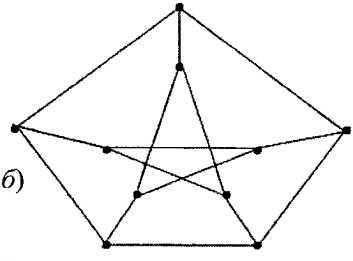


Рис. 5.1.4

3. Повністю незв'язні графи.

Визначення 116. *Граф, у якого множина ребер пуста, називається* **повністю незв'язним** *або* **пустим**.

Будемо позначати повністю незв'язні графи з n вершинами через N_n. На рис. 5.1.5 зображено граф N_4.

Рис. 5.1.5. Пустий граф

Зауважимо, що кожний пустий граф є регулярним графом степеня 0.

4. Платонові графи.

Визначення 117. *Платоновими графами називаються графи, утворені вершинами і ребрами п'яти правильних многогранників — платонових тіл: тетраедра, куба, октаедра, додекаедра і ікосаедра.*

Граф K_4, зображений на рис. 5.1.2, відповідає тетраедру, а графи, що відповідають кубу і октаедру, показані на рис. 5.1.4 (*а*) і 5.1.6 відповідно.

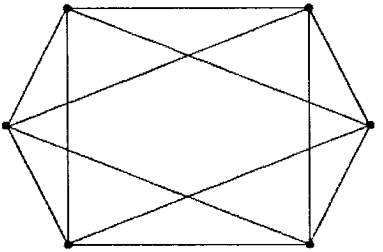


Рис. 5.1.6

Решту графів, які відповідають додекаедру і ікосаедру, пропонується побудувати читачеві (див. вправу в кінці параграфа).

5. Двочасткові графи.

Визначення 118. *Граф називається* **двочастковим**, *якщо існує таке розбиття множини його вершин на два класи, при якому кінці кожного ребра лежать у різних класах.*

Двочастковий граф можна визначити іншим шляхом — у термінах розфарбування його вершин двома кольорами, наприклад червоним і синім. При цьому граф називається двочастковим, якщо кожну його вершину можна пофарбувати синім або червоним кольором, так, щоб кожне його ребро мало один кінець червоний, а другий — синій.

Якщо у двочастковому графі дві довільні вершини з різних класів суміжні, то такий граф називається **повним двочастковим графом**. Повний двочастковий граф, у якого один клас має m вершин, а другий — n вершин, позначають $K_{m,n}$.

Повний двочастковий граф виду $K_{1,n}$ називається **зірковим графом**. На рис. 5.1.7 зображений граф $K_{1,5}$.

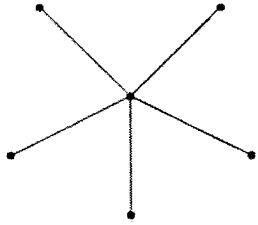


Рис. 5.1.7. Зірковий граф

Аналогічно можна ввести k-часткові графи. Граф називається **k-частковим графом**, якщо існує таке розбиття множини його вершин на k класів, при якому довільне ребро графа з'єднує дві вершини з різних класів.

6. Орієнтовані графи (орграфи)

Визначення 119. *Граф* $G = (V, E)$ *називається* **орієнтованим графом (орграфом)**, *якщо* $E \subseteq V^2$, *тобто вершини всіх його ребер упорядковані. Якщо* $(u, v) \in E$, *то вершину* u *називають* початковою вершиною, *а* v — кінцевою вершиною *ребра.*

Орграфи зображуються так само як і графи, з тією лише різницею, що їх ребра позначаються стрілками, які ведуть з початкової вершини ребра в кінцеву. Іншими словами, якщо (u, v) — ребро орграфа, то з вершини u у вершину v веде стрілка. Приклади орграфів показані на рис. 5.1.8.

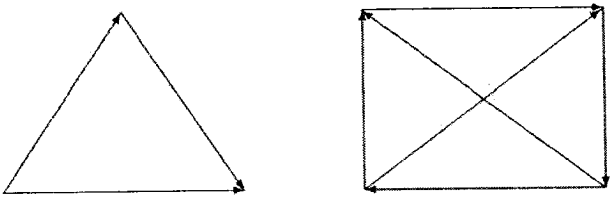


Рис. 5.1.8. Орграфи

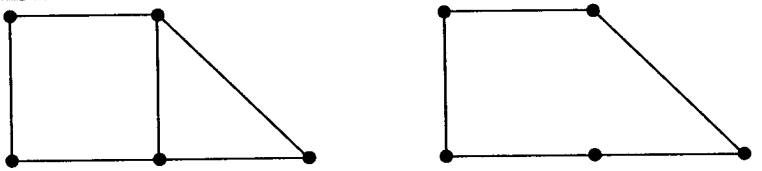
Можна визначити і орієнтований мультиграф. Граф $G = (V, E)$ називається **орієнтованим мультиграфом**, якщо $E \subseteq MV^2$ і кожне його ребро упорядковане, тобто вказано, яка вершина перша, а яка друга. Отже, у орграфах ребра (u, v) і (v, u) різні.

5.1.3. Ізоморфізм графів. Підграфи

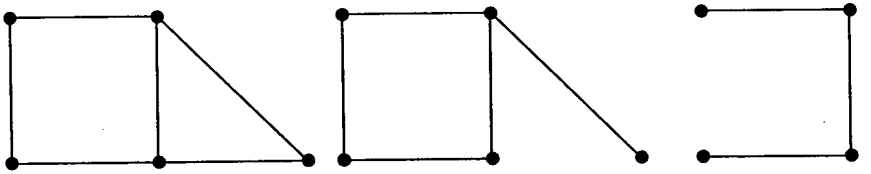
Нехай $G = (V, E)$ і $H = (V_1, E_1)$ — графи і $h : V \to V_1$ — взаємно однозначна відповідність (тобто $|V| = |V_1|$). Відображення h називається **ізоморфізмом графів** G і H, якщо для довільних вершин u і v графа G їх образи $h(u)$ і $h(v)$ суміжні у графі H тоді і тільки тоді, коли u і v суміжні в G. Якщо таке відображення h існує, то графи G і H називаються **ізоморфними**. Очевидно, що відношення ізоморфізму графів є відношенням еквівалентності. Ізоморфні графи, як правило, не розрізняють між собою.

Граф $H = (V', E')$ називається **підграфом графа** $G = (V, E)$, якщо $V' \subseteq V$ і $E' \subseteq E$. Якщо H підграф графа G, то говорять, що H знаходиться у графі G. Підграф H графа G називається **остовним підграфом**, коли $V' = V$.

Приклад 5.1.1



Граф і його остовний підграф



Граф і його підграфи ♠

5.2. Контрольні питання, задачі і вправи

Контрольні питання

1. Що називається а) графом, б) мультиграфом, в) псевдографом?
2. Яка різниця між графом і псевдографом?
3. Який граф називається а) регулярним, б) регулярним степеня k, в) повним?
4. Чи буде а) повний граф регулярним графом, б) регулярний граф повним?
5. Чи буде скінченним граф, у якого скінченне число а) вершин, б) ребер?
6. Які ви знаєте різновиди графів?
7. Побудуйте платонові графи для ікосаедра і додекаедра.
8. Чи будуть платонові графи регулярними?

5.3. Операції над графами

На практиці часто зустрічаються графи, які будуються із деякого висхідного графа за допомогою вилучення однієї з його вершин або одного з його ребер. Існують і інші можливі перетворення графів, які визначаються як операції над графами. Розглянемо основні операції над графами і деякі їх властивості.

1. Операція вилучення ребра. Нехай $G = (V, E)$ — граф і $e \in E$ — деяке його ребро. Говорять, що граф $G_1 = G - e$ одержаний з графа G в результаті операції вилучення ребра e, якщо $G_1 = (V, E \setminus \{e\})$. Отже, кінці ребра e не вилучаються з множини V.

Неважко показати, що для довільних ребер e і e_1 графа G має місце така тотожність:

$$(G - e) - e_1 = (G - e_1) - e.$$

Дійсно, оскільки має місце тотожність $(A \setminus B) \setminus C = A \setminus (B \cup C)$ (див. вправу 13 із розділу 1), то маємо

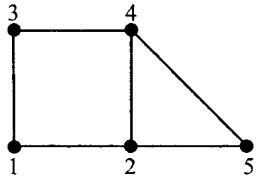
$$G_1 = G - e = (V, E \setminus \{e\})$$

і

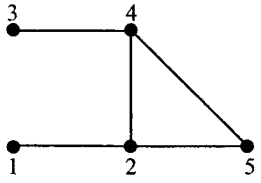
$$(G - e) - e_1 = G_1 - e_1 = (V, (E \setminus \{e\}) \setminus \{e_1\}) = (V, E \setminus (\{e\} \cup \{e_1\})) =$$
$$= (V, E \setminus (\{e_1\} \cup \{e\})) = (V, (E \setminus \{e_1\}) \setminus \{e\}) = (G - e_1) - e.$$

Отже, якщо виконується підряд кілька операцій вилучення ребра, то результат не залежить від порядку, в якому ці ребра вилучаються з графа.

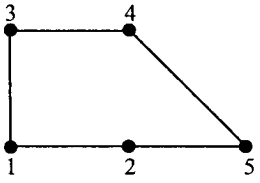
Приклад 5.3.2



Граф G



Граф $G \setminus \{(1, 3)\}$

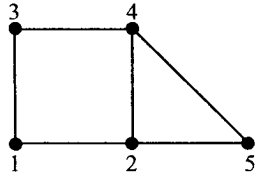


Граф $G \setminus \{(2, 4)\}$ ♠

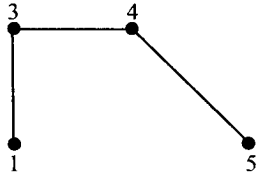
2. Операція вилучення вершини. Нехай $G = (V, E)$ *і* $v \in V$. Говорять, що граф $G_1 = G - v$, одержаний з графа G в результаті операції вилучення вершини v, якщо вершина v вилучена із V, а із E вилучені всі ребра, інцидентні з вершиною v.

Неважко переконатися, що операція вилучення вершини не залежить від порядку, в якому вилучаються вершини з графа.

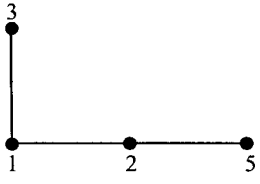
Приклад 5.3.3



Граф G



Граф $G - \{2\}$



Граф $G - \{4\}$ ♠

Операції вилучення ребра, вершини і переходу до підграфа — це операції, за допомогою яких можна із висхідного графа отримувати інші графи з меншим числом вершин і ребер. Є й інші операції, які дають можливість будувати з висхідних графів нові графи, з більшим числом вершин і ребер.

3. Операція введення ребра. Якщо $u, v \in V$ і $(u, v) \notin E$ у графі $G = (V, E)$, то визначається граф $G + e = (V, E \cup \{e\})$, де $e = (u, v)$, про який говорять, що він отриманий у результаті операції введення ребра.

В силу комутативності операції об'єднання множин, можна стверджувати, що послідовність операцій введення ребер у граф G не залежить від порядку, в якому ці ребра вводяться в граф G. Іншими словами, має місце тотожність

$$(\forall e, e_1 \in E)\,((G + e) + e_1 = (G + e_1) + e).$$

4. Операція введення вершини в ребро. Нехай (u, v) — деяке ребро графа G. Введенням вершини w в ребро (u, v) називається операція, у результаті якої отримуємо два ребра виду (u, w) і (w, v), а ребро (u, v) при цьому вилучається з графа G.

5. Операція об'єднання графів. Граф F називається *об'єднанням графів* $G = (V, E)$ і $H = (V_1, E_1)$, якщо $F = (V \cup V_1, E \cup E_1)$. Граф F позначається через $G \cup H$. Об'єднання графів $F = G \cup H$ називається диз'юнктивним, якщо $V \cap V_1 = \varnothing$.

Безпосередньо з визначення операції об'єднання графів випливає, що

$$(\forall G, H)\,(G \cup H = H \cup G)$$

Операція диз'юнктивного об'єднання графів дає можливість ввести до розгляду ще один важливий тип графів.

Визначення 120. *Граф називається* **зв'язним**, *якщо його не можна зобразити у вигляді диз'юнктивного об'єднання його підграфів, і* **незв'язним** *у протилежному випадку.*

Отже, довільний незв'язний граф можна зобразити у вигляді диз'юнктивного об'єднання скінченного числа зв'язних підграфів. Кожний з таких зв'язних підграфів називається **компонентом зв'язності**.

Зв'язний регулярний граф степеня 2 називається **циклічним графом**. Циклічний граф з n вершинами позначається C_n. Циклічний граф C_6 наведено на рис. 5.3.9.

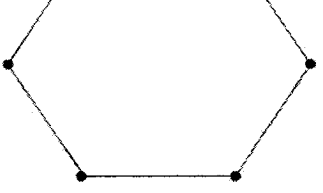
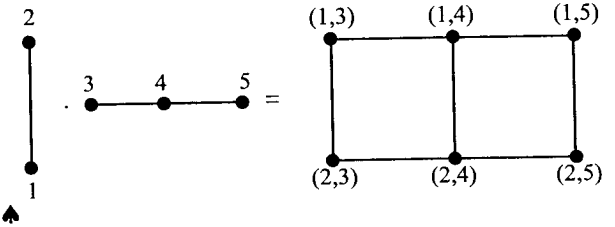


Рис. 5.3.9. Циклічний граф

6. Добуток графів. Добутком графів $G = (V, E)$ і $H = (V_1, E_1)$ називається граф $F = G \times H$, у якого $V = V \times V_1$, а E визначається таким чином:

вершини (u, u_1) і (v, v_1) суміжні в F тоді і тільки тоді, коли $u = v$, а u_1 і v_1 суміжні в H або $u_1 = v_1$, а u і v суміжні в G.

Приклад 5.3.4



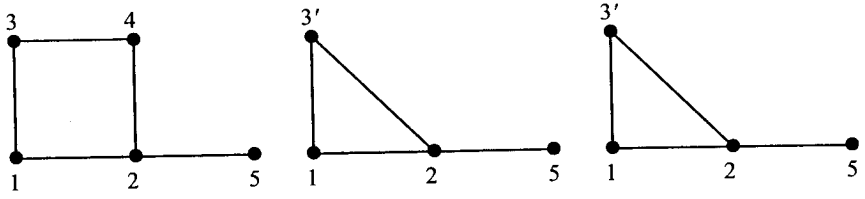
♠

7. Ототожнення (злиття) вершин. Якщо $G = (V, E)$ — граф, u, v — дві його вершини і См(u) = $\{u_1,..., u_k\}$, См(v) = $\{v_1,..., v_l\}$, то граф $H = G - u - v$, отримано приєднанням нової вершини u' до множини вершин H і множини ребер вигляду (u', u_i), (u', v_j) (i = 1, 2,..., k, j = 1, 2,..., 1) до множини ребер H, називається графом, отриманим із G шляхом ототожнення вершин u і v.

8. Операція стягування ребра (u, v) у графі $G = (V, E)$ означає ототожнення вершин u і v у графі G.

Операція стягування ребра дозволяє ввести таке поняття. Граф G називається графом, який **стягується до графа** G, якщо H можна отримати з G, за допомогою деякої послідовності операцій стягування ребра. Легко помітити, наприклад, що граф Петерсена стягується до графа K_5 і, отже, до довільного графа K_n, де $n < 5$. Очевидно також, що довільний непустий зв'язний граф, відмінний від K_1, стягується до K_2. Але вже не довільний зв'язний граф стягується до K_3. Наприклад, простий ланцюг P_n не стягується до K_3. Логічно ввести параметр $\xi(G)$ — максимум порядків повних графів, до яких стягується граф G. Параметр $\xi(G)$ називається *числом Хадвігера графа G*.

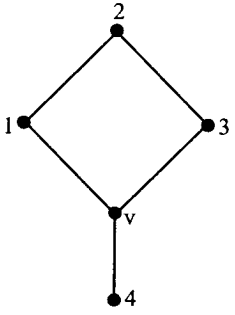
Приклад 5.3.5

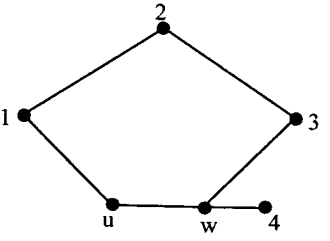


Граф G Ототожнення 3 і 4 в G Стягування 3 і 4 в G ♠

9. Операція роздвоєння (розщеплення) вершини. Нехай v — деяка з вершин графа G. Розіб'ємо множину суміжних з нею вершин довільним чином на дві частини M і P, а потім виконаємо таке перетворення графа G: вилучимо вершину v разом з інцидентними їй реб-

рами і введемо дві нових вершини u і w разом з ребром, яке з'єднує ці вершини, вершину u з'єднаємо ребром з кожною вершиною множини M, а вершину w — з кожною вершиною із множини P. Отриманий у результаті граф позначимо символом $G^{\sim}$ і будемо вважати, що він отриманий із графа G в результаті роздвоєння (розщеплення) вершини v (див. наведений нижче рисунок).

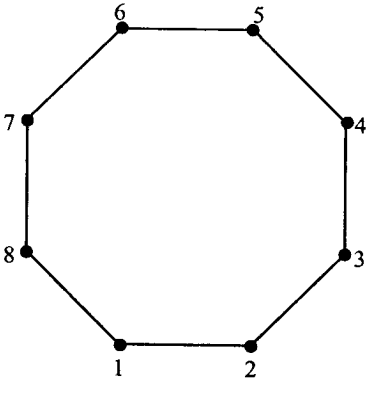


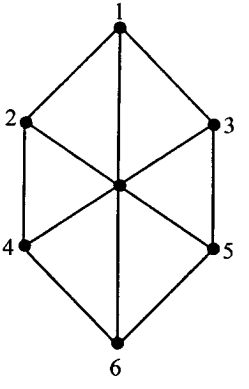


10. Операція з'єднання графів. Нехай $G = (V, E)$ і $G_1 = (V_1, E_1)$ — два графи, у яких множини вершин V і V_1 не перетинаються, тобто $V \cap V_1 = \varnothing$. **Операція з'єднання графів** G і G_1 полягає в тому, що множини V і V_1 об'єднуються, а потім з'єднуються ребрами кожна вершина графа G з кожною вершиною графа G_1. Тобто, якщо E^u означає множину ребер, яка одержана об'єднанням множин E і E_1 разом з утвореними новими ребрами, то $G = G + G_1 = (V \cup V_1, E^u)$.

Зрозуміло, що операція з'єднання графів може бути виражена у вигляді добутку (суперпозиції) операції об'єднання графів G і G_1 і послідовності операцій введення ребра.

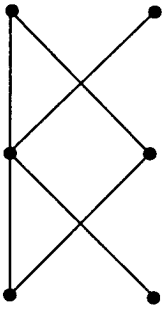
З'єднання графів N_1 і C_{n-1} ($n > 3$) називається **колесом з n вершинами** і позначається через W_n. Нижче зображено графи C_8 і W_7.



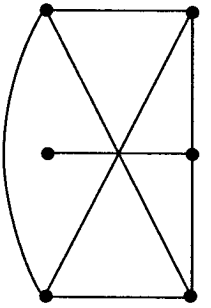


11. Операція доповнення графа. Нехай $G = (V, E)$ — граф. **Доповненням** G^* графа G називається граф з множиною вершин V, в якому дві вершини суміжні тоді і тільки тоді, коли вони не суміжні в графі G. Звідси випливає, що коли граф G має n вершин, то граф G^* можна побудувати, вилучивши з графа K_n усі ребра, які належать G (граф G вважається підграфом графа K_n). Очевидно також, що доповненням повного графа є пустим графом, і навпаки, доповнення пустого графа є повним графом. Неважко довести, що доповнення регулярного графа є регулярним графом.

Приклад 5.3.6



Графа G



Доповнення графа G

5.4. Властивості графів

Розглянемо детальніше деякі властивості графів і зв'язаних з ними понять.

5.4.1. Маршрути, цикли, зв'язність

Маршрутом в заданому графі $G = (V, E)$ називається скінченна послідовність його ребер вигляду

$$(v_0, v_1), (v_1, v_2),\ldots,(v_{k-1}, v_k).$$

Число k ребер маршруту називається **довжиною** цього маршруту. Часто можна зустріти й таке визначення маршруту в графі: послідовність вершин $v_0, v_1, \ldots, v_k$ графа $G = (V, E)$ називається **маршрутом**, який з'єднує вершини v_0 і v_k, якщо $(v_i, v_{i+1}) \in E$, $i = 0, 1,..., k - 1$. В обох випадках вершини $v_0, v_1,..., v_k$ називаються вершинами маршруту.

Очевидно, що відношення «маршрут, який з'єднує вершини...» є симетричним і транзитивним відношенням.

Маршрут називається **ланцюгом**, якщо всі його ребра різні, і **простим ланцюгом**, якщо всі його вершини крім, можливо, першої і останньої різні. Маршрут називається **циклічним**, якщо перша і остання його вершини збігаються.

Циклом називається циклічний ланцюг, а **простим циклом** — простий циклічний ланцюг.

Граф називається **ациклічним графом** або **лісом**, якщо в ньому відсутні цикли. Безпосередньо з визначення маршруту і циклу випливають такі твердження.

Твердження 11. *Довільний маршрут, який з'єднує будь-які дві вершини графа, має простий ланцюг, що з'єднує ці вершини.*

Твердження 12. *Довільний цикл у графі має простий цикл.*

У термінах маршрутів тепер можна дати інше визначення зв'язного графа.

Визначення 121. *Граф називається* **зв'язним**, *якщо дві довільні його вершини зв'язані маршрутом (а в силу попереднього твердження, можна сказати і простим ланцюгом). Зв'язний підграф H графа G називається* **максимальним**, *якщо H не знаходиться ні в якому зв'язному підграфі графа G. Максимальний зв'язний підграф графа називається* **компонентом зв'язності**.

Наступне твердження встановлює еквівалентність двох визначень зв'язного графа.

Теорема 108. *Граф зв'язний тоді і тільки тоді, коли його не можна представити у вигляді диз'юнктивного об'єднання двох графів.*

Доведення. Нехай граф зв'язний, тобто дві довільні його вершини u і v зв'язані маршрутом. Припустимо, що граф G являє собою диз'юнктивне об'єднання двох підграфів, а вершини u і v належать різним підграфам. Тоді довільний простий ланцюг, який з'єднує вершини u і v, повинен мати ребро, інцидентне деяким двом вершинам з різних підграфів. Але такого ребра не існує в силу нашого припущення.

Припустимо тепер, що граф G не можна зобразити у вигляді об'єднання двох підграфів і не існує ніякого простого ланцюга, який з'єднує задану пару вершин u і v. Тоді вершини u і v належать різним компонентам зв'язності. А це означає, що граф G можна зобразити у вигляді об'єднання двох підграфів, одним з яких є компонент зв'язності, якому належить вершина u, а другим — об'єднання решти компонент. ■

5.4.2. Властивості регулярних графів

Нехай G — регулярний граф степеня k. Степінь регулярного графа позначають $deg(G)$. Очевидним наслідком визначення регулярного графа є таке твердження.

Твердження 13. *Повний граф є регулярним графом.*

Користуючись лемою про рукостискання, неважко встановити справедливість такого простого твердження для регулярних графів.

Твердження 14. *Не існує регулярного графа з n вершинами степеня k, у якого k і n обоє непарні.*

Дійсно, якщо n і k — непарні числа, то їх добуток $n \cdot k$ теж непарне число, але $n \cdot k = 2|E|$ в силу леми про рукостискання. Отримана суперечність доводить наше твердження.

Теорема 109. *Нехай n, $d \in N$ — натуральні числа, одне з яких парне і для яких виконується нерівність $0 \le d \le n - 1$. Тоді існує регулярний граф порядку n і степеня d.*

Доведення. Для $d = 0$ твердження очевидне. Крім того, якщо G — регулярний граф порядку n і степеня d, то його доповнення — граф G^* — теж регулярний граф і $degG^* = n - 1 - d$. У зв'язку з цим достатньо розглянути випадок, коли $0 < d \le (n-1)/2$.

Нехай Z_n — адитивна група лишків цілих чисел за модулем n, $0 \notin A$ і для $x \in A$ x-клас також належить множині A. Визначимо граф G порядку n з множиною вершин Z_n такою умовою: вершини x і y суміжні, якщо $x - y \in A$. Очевидно, що граф G регулярний і степінь його дорівнює $|A|$. Залишається лише довести, що для довільного числа d, яке задовольняє умови теореми, існує відповідна d-елементна множина A. При $d = 2k$ можна взяти $A = \{\pm1, \pm2, ..., \pm k\}$, а при $d = 2k + 1$ число n парне і можна взяти $A = \{\pm1, \pm2, ..., \pm k, n/2\}$ (див. рис. 5.4.10, де $n = 8$, $d = 3$). ■

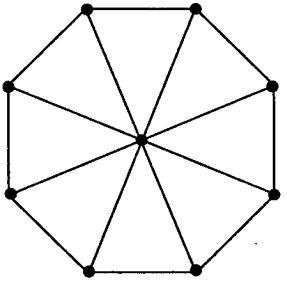


Рис. 5.4.10

Теорема 110. *Нехай $G = (V \cup V_1, E)$ — непустий регулярний двочастковий граф, тоді $|V| = |V_1|$, де V, V_1 — класи розбиття множини вершин графа G.*

Доведення. Оскільки множині V належить лише один кінець кожного з ребер графа G, то число m його ребер дорівнює $|V| \cdot deg(G)$. Аналогічно $m = |V_1| \cdot deg(G)$. Отже, $|V| \cdot deg(G) = |V_1| \cdot deg(G)$. В силу того, що G непустий, то $deg(G) \neq 0$ і, отже, $|V| = |V_1|$.

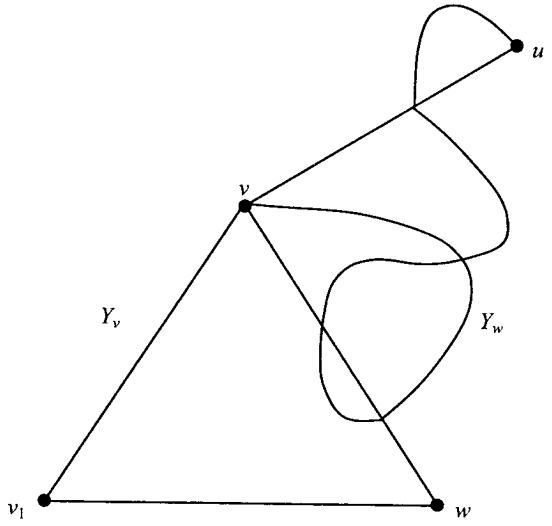
5.4.3. Властивості двочасткових графів

Існує простий критерій двочастковості графа, який виражається в термінах довжини циклів.

Теорема 111 *(Кьоніг). Граф $G = (V, E)$ є двочастковим тоді і тільки тоді, коли він не має циклів непарної довжини.*

Доведення. Нехай G — двочастковий граф і C — один із його циклів довжини k. Пройдемо всі ребра цього циклу в тій послідовності, в якій вони зустрічаються в ньому, починаючи з деякої вершини u. Пройшовши k ребер, повертаємося знову до вершини u. Оскільки кінці кожного ребра лежать у різних частинах, то k — парне число. Нехай тепер G — зв'язний граф n-го порядку ($n > 1$) не має циклів непарної довжини і $u \in V$. Побудуємо розбиття $V = V_1 \cup V_2$ таким чином: довільну вершину v графа G включимо до класу V_1, якщо відстань $d(v, u)$ — парне число, і включимо її до класу V_2, якщо ця відстань — непарне число. Покажемо тепер, що підграфи $G_1 = (V_1, E_1)$ і $G_2 = (V_2, E_2)$, які утворюються множинами вершин V_1 і V_2 відповідно, є пустими (тобто вершини із V_1 і V_2 не з'єднані між собою жодним ребром). Припустимо, що це не так, що існують суміжні вершини v і w, які належать одному й тому самому класу. Тоді жодна з них не збігається з вершиною u. Нехай P найкоротший ланцюг, який з'єднує вершини v і w, а v — остання вершина, якщо відлік вести від вершини v, спільна для ланцюгів P і Q, яка належить ланцюгу P (див. наведений нижче рисунок). Позначимо через X_v і Y_v відповідно підланцюги, які з'єднують u і v і v_1 і v вершини ланцюга P, а через X_w і Y_w — відповідно підланцюги ланцюга Q, які з'єднують вершини u і v та v_1 і w.

Очевидно, що довжини ланцюгів Y_v і Y_w нерівні і, отже, довжини ланцюгів Y_v і Y_w або парні, або непарні. Але тоді об'єднання ланцюгів Y_v і Y_w і ребра (v, w) є циклом непарної довжини. Отримана суперечність доводить теорему. ■



Наслідок 27. *Граф двочастковий тоді і тільки тоді, коли він не має простих циклів непарної довжини.*

З доведення попередньої теореми випливає простий спосіб розпізнавання двочастковості графа. Він ґрунтується на простому прийомі. Будемо приписувати номери 0 і 1 вершинам графа G таким чином:

— починаючи з довільної вершини u графа G, приписуємо їй номер 0;

— кожній вершині із множини См(u) приписуємо номер 1;

— для всіх вершин, суміжних з вершинами множини См(u), приписуємо номер 0;

— для всіх вершин, які одержали номер 0, знаходимо всі суміжні з ними вершини і приписуємо їм номер 1 і т. д.

Після того як усі вершини будуть перенумеровані, будуємо дві множини V_0 і V_1, які відповідно включають усі вершини з номерами 0 і номерами 1. Якщо графи $G_1 = (V_0, E_0)$ і $G_2 = (V_1, E_1)$ пусті, то граф G двочастковий, а якщо ні (тобто або E_0, або E_1 непуста), то G не є двочастковим.

5.4.4. Властивості зв'язних графів

З'ясуємо тепер деякі властивості зв'язних графів. Насамперед встановимо такий факт.

Теорема 112. *Довільний граф $G = (V, E)$ єдиним чином зображується у вигляді диз'юнктивного об'єднання своїх компонентів зв'язності.*

Доведення. Визначимо на множині вершин графа G відношення R : $uRv \Leftrightarrow (u = v)$ або існує маршрут у G, який з'єднує вершини u і v.

Неважко перевірити, що відношення R є відношенням еквівалентності. Отже, множина вершин графа розбивається на класи, що не перетинаються. Нехай $V = \bigcup_{i=1}^{k} V_i$ — це розбиття, тоді підграфи $G_i = (V_i, E_i)$, де E_i — множина ребер у графі G, що з'єднують вершини із V_i у графі G, і тільки вони є компонентами зв'язності графа G і $G = \bigcup_{i=1}^{k} C_i$ — диз'юнктивне об'єднання. ■

Часто виникає питання про число ребер зв'язного графа.

Нехай $G = (V, E)$ — граф з n вершинами і k компонентами зв'язності. Якщо такий граф зв'язний, то природно чекати, що число ребер у ньому мінімальне, коли він ациклічний, і максимальне, коли він повний. Звідси випливає така оцінка для числа ребер зв'язного графа:

$$n - 1 \le |E| \le n \cdot (n - 1)/2.$$

Насправді має місце більш сильний результат.

Теорема 113. *Нехай G — граф з n вершинами і k компонентами зв'язності. Тоді число m його ребер задовольняє нерівностям*

$$n - k \le m \le (n - k) \cdot (n - k + 1)/2.$$

Доведення. Нерівність $m \ge n - k$ легко довести методом математичної індукції. Дійсно, якщо G — повністю незв'язний граф, то нерівність справедлива: у цьому випадку $k = n$, $m = 0$ і $0 \ge n - k = 0$. Нехай G має мінімальне число ребер, наприклад m', тоді вилучення одного ребра приводить до збільшення компонентів зв'язності на одиницю. Тобто, отриманий граф буде мати n вершин, $k + 1$ компонентів зв'язності і $m' - 1$ ребро. В силу індуктивного припущення маємо

$$m' - 1 \ge n - (k + 1)$$

або

$$m' \ge n - k,$$

що й потрібно було довести.

При доведенні справедливості верхньої оцінки можна вважати, що кожен компонент зв'язності графа G є повним графом. Припустимо, що C_i і C_j — два компонети зв'язності відповідно з n_i і n_j вершинами, де $n_i \ge n_j > 1$. Якщо замінити C_i і C_j на повні графи з $n_i + 1$ і

$n_j - 1$ вершинами, то загальна кількість вершин не зміниться, а число ребер збільшиться на деяку додатну величину.

$$1/2\{((n_i + 1) \cdot n_i - n_i \cdot (n_i - 1) - (n_j - 1) \cdot n_j + (n_j - 1) \cdot (n_j - 2)\} =$$
$$= 1/2\{ n_i^2 + n_i - n_i^2 + n_i - n_j + n_j + n_j - 3n_j + 2\} = 1/2(2n_i - 2n_j + 2) =$$
$$= n_i - n_j + 1.$$

Отже, для того щоб число ребер у графі G було максимальним при заданих n і k, граф G має складатися з $k - 1$ ізольованих вершин і повного графа з $n - k + 1$ вершинами. А звідси відразу випливає потрібна нерівність. ■

Наслідок 28. *Довільний граф порядку n, який має більше ніж* $(n - 1)(n - 2)/2$ *ребер, зв'язний.*

Дійсно, у цьому випадку число компонентів зв'зності такого графа має бути строго менше 2-х. Отже $k = 1$, тобто граф зв'язний.

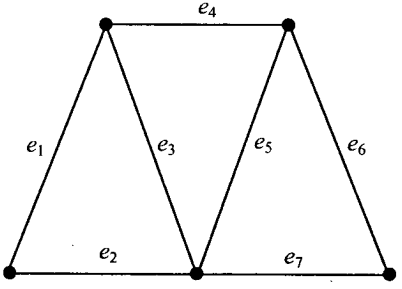
Друге питання, яке теж часто виникає, — це питання про те, наскільки сильно зв'язним є зв'язний граф. Іншими словами це питання можна сформулювати так: скільки потрібно вилучити ребер з графа, щоб він перестав бути зв'язним?

Для відповіді на це питання введемо деякі визначення загального характеру.

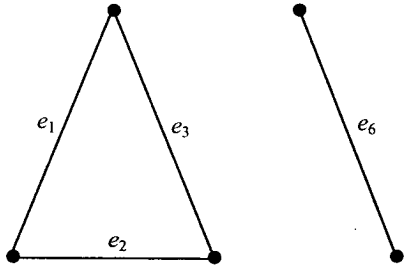
Множина ребер довільного графа G, яка називається **множиною, що розділяє граф G**, — це така множина його ребер, вилучення якої з графа G призводить до збільшення числа його компонентів зв'язності. Якщо граф G зв'язний, то множиною ребер G, що розділяє граф G, називається така множина його ребер, вилучення якої з графа G приводить до незв'язного графа.

Приклад 5.4.7

У наведеному нижче графі G, множина ребер $\{e_4, e_5, e_7\}$ є множиною, що розділяє граф G.



Граф G



Результат вилучення ребер $\{e_4, e_5, e_7\}$

Очевидно, що $\{e_6, e_7\}$, $\{e_5, e_6, e_7\}$, $\{e_1, e_2, e_3\}$, $\{e_1, e_3, e_4\}$ і т. д. — множини ребер, які розділяють граф G. ♠

Отже, з наведеного прикладу випливає, що множина ребер, яка розділяє граф, може мати власну підмножину, котра також розділяє граф.

Розрізом графа G називається така множина ребер, що розділяє граф G, жодна власна підмножина якої не є множиною, що розділяє цей граф. У розглянутому вище прикладі розрізами будуть множини $\{e_4, e_5, e_7\}$, $\{e_6, e_7\}$, $\{e_1, e_3, e_4\}$.

Розріз графа G, який складається лише з одного ребра, називається **мостом**. Наприклад, якщо граф G є колесом, то в ньому довільний його розріз буде мостом.

Зауважимо, що в результаті вилучення ребер, які входять до складу розрізу графа G, число компонентів зв'язності графа G збільшується рівно на 1.

Граф, ізоморфний своєму доповненню, називається графом, що доповнює сам себе. Прикладами графів, які доповнюють самі себе, можуть служити графи: K_1 — повний граф порядку 1, C_5 — простий цикл довжини 5 і т. д. Очевидно також, що коли G ізоморфний H, то G^* ізоморфний H^* і $G^{**} = G$.

Теорема 114. *Для довільного графа G або він сам, або його доповнення G^* є зв'язним.*

Доведення. Нехай $G = (V, E)$ — незв'язний граф, A — один із його компонентів зв'язності і $B = V \setminus A$. Тоді для довільних вершин u із A і v із B у графі G^* є маршрут довжини 1, оскільки ці вершини зв'язані ребром (u, v). Отже, довільна вершина із B зв'язана з вершиною u маршрутом довжини 1, а довільна вершина із A зв'язана з u маршрутом довжини не більше ніж 2. Тобто довільна вершина u зв'язана з довільною вершиною v маршрутом. Отже, G^* — зв'язний граф. ■

Теорема 115 *Нехай $G = (V, E)$ — зв'язний граф і $e \in E$ — деяке його ребро. Тоді якщо e належить якому-небудь циклу, то $G - e$ — зв'язний. Якщо ж e не належить ні одному циклу, то граф $G - e$ має рівно два компоненти зв'язності.*

Доведення. Нехай $e = (u, v)$ належить деякому циклу C графа G. Виконаємо заміну в кожному ланцюгові, який з'єднує вершини x і y, і включає ребро e, ланцюгом $C - e$. Отримаємо маршрут з вершини x до вершини y, який не має ребра e. Тобто у графі G довільні дві вершини, які не збігаються між собою, з'єднані маршрутом. А це означає, що $G - e$ — зв'язний.

Нехай тепер $e = (u, v)$ не входить ні до якого циклу графа G. Тоді очевидно, що вершини u і v належать різним компонентам зв'язності, наприклад: G_u і відповідно G_v графа $G - e$. Для довільної вершини $x \neq u$ в G існує маршрут із x в u в силу зв'язності G. Якщо e в цей маршрут не входить, то $x \in G_v$. Тобто G_u і G_v — два компоненти зв'язності. ■

Ці теореми дають деяку характеристику операціям вилучення ребра по відношенню до властивості зв'язності. Зрозуміло, що обернена операція — операція введення ребра — не порушує цієї властивості.

5.4.5. Метричні характеристики зв'язних графів

Нехай $G = (V, E)$ — зв'язний граф, а u і v — дві його вершини, не рівні між собою.

Відстанню між вершинами u і v називається довжина найкоротшого маршруту, який з'єднує вершини u і v, і позначається $d(u, v)$. Припустимо також, що $d(u, u) = 0$. Очевидно, що для введеної таким чином відстані, мають місце такі аксіоми:

1) $d(u, v) \geq 0$;
2) $d(u, v) = 0 \quad \Leftrightarrow u = v$;
3) $d(u, v) = d(v, u)$;
4) $d(u, v) + d(v, w) \geq d(u, w)$ (ii).

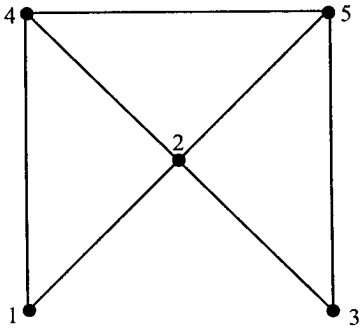
Поняття відстані між вершинами дає можливість визначити поняття степеня графа. Нехай G — зв'язний граф, k — натуральне число. Граф G^k — k-та, степінь графа G, має ту саму множину вершин, що й G, а нерівні між собою вершини u і v сумісні у графі G^k тоді і тільки тоді, коли $d(u, v) \leq k$. Безпосередньо з визначення випливає така властивість графа G^k.

Твердження 15. *Якщо $k \geq |V| - 1$, де V — множина вершин графа G, то G^k — повний граф.*

Нехай u деяка фіксована вершина графа $G = (V, E)$. Величина $e(u) = \max d(u, v)$, $v \in V$ називається *ексцентриситетом вершини u*. Максимальний серед усіх ексцентриситетів вершин графа G називається *діаметром графа G*, і позначається $d(G)$. Отже, $d(G) = \max e(u)$, $u \in V$.

Вершина v графа G називається *периферійною*, якщо $e(v) = d(G)$. Простий ланцюг довжини $d(G)$, відстань між початковою і кінцевою вершинами якого дорівнює $d(G)$, називається *діаметральним ланцюгом*.

Приклад 5.4.8
Дано граф:



Граф G

У даному графі G маємо $d(1, 2) = 1$, $d(1, 3) = 2 = d(1, 5)$, $e(1) = 2$, $e(2) = 1$, $d(G) = 2$. Усі вершини, крім вершини 2, периферійні, а $(1, 2, 3)$ — діаметральний ланцюг. ♠

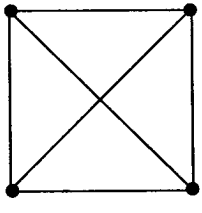
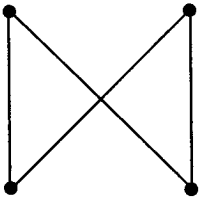
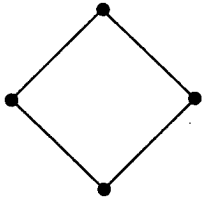
5.4.6. Властивості ейлерових графів

Одним з важливих різновидів зв'язних графів є так звані ейлерові і гамільтонові графи.

Визначення 122. *Зв'язний граф G називається ейлеровим, якщо існує замкнутий ланцюг, який включає кожне його ребро. Такий ланцюг називається ейлеровим ланцюгом.*
Зв'язний граф G називається напівейлеровим, якщо в ньому існує ланцюг, який включає кожне його ребро.

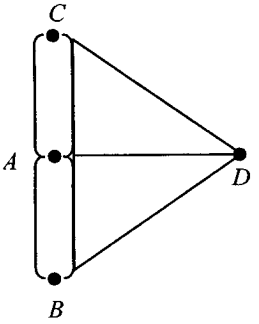
Таким чином, довільний ейлеровий граф буде напівейлеровим.

Приклад 5.4.9. Наведені нижче графи відповідають неейлеровому, ейлеровому і напівейлеровому графам.

♠

Поняття ейлерового графа виникло у зв'язку з відомою головоломкою про кенігсбергські мости, в якій необхідно було дізнатися, чи має граф, зображений нижче на рисунку, ейлерів ланцюг, чи ні.



Виникає природне запитання: як встановити, що заданий граф є ейлеровим? Відповідь на це запитання дають такі твердження.

Теорема 116 *Якщо степінь кожної вершини скінченного графа не менше двох, то граф має цикл.*

Доведення. Нехай v — довільна вершина графа G. Користуючись методом побудови по індукції, побудуємо маршрут $v, v_1,..., v_k,...$ так, що v_{i+1} — суміжна з v_i і відмінна від v_{i-1}. Існування такої вершини випливає з умов теореми. Оскільки граф G скінченний, то на деякому кроці побудови нашого маршруту прийдемо до вершини, яку вже було вибрано раніше. Нехай v_k — перша з таких вершин, тоді частина маршруту, яка лежить між першою і другою появою v_k у цьому маршруті, і є шуканим циклом. ■

Зауважимо, що дана теорема справедлива для псевдографа (який має петлі) і для мультиграфа (який має кратні ребра).

Теорема 117. *Зв'язний граф G є ейлеровим тоді і тільки тоді, коли кожна вершина G має парний степінь.*

Доведення. Нехай G має ейлеровий ланцюг P, тоді при довільному проходженні ланцюга P через довільну із вершин графа G степінь цієї вершини збільшується на 2. А оскільки кожне ребро зустрічається в P тільки один раз, то кожна вершина повинна мати парний степінь.

Доведення у зворотному напрямку проводиться індукцією за числом ребер у графі G.

Наслідок 29. *Зв'язний граф ейлеровий тоді і тільки тоді, коли множину його ребер можна розбити на цикли, які не перетинаються.*

Наслідок 30. *Зв'язний граф напівейлеровий тоді і тільки тоді, коли він має не більше двох вершин непарного степеня.*

Доведення. Нехай v — довільна вершина графа G. Користуючись методом побудови по індукції, побудуємо маршрут $v, v_1,..., v_k,...$ так, що v_{i+1} — суміжна з v_i і відмінна від v_{i-1}. Існування такої вершини випливає з умов теореми. Оскільки граф G скінченний, то на деякому кроці побудови нашого маршруту прийдемо до вершини, вибраної раніше. Нехай v_k — перша з таких вершин, тоді частина маршруту, яка лежить між першою і другою появою v_k у цьому маршруті, і є шуканим циклом.

Наслідок доведено.

Зауважимо, що дана теорема справедлива для псевдографів (які мають петлі) і для мультиграфів (які мають кратні ребра).

Слід також зауважити, що коли напівейлерів граф має рівно дві вершини непарного степеня, то довільний напівейлеровий ланцюг (сенс поняття очевидний) буде обов'язково мати одну з цих вершин початковою, а другу — кінцевою. З леми про рукостискання випливає, що граф не може мати тільки одну вершину непарного степеня.

Теорема 118 **(Флері).** *Нехай $G = (V, E)$ — ейлеровий граф, тоді наступна процедура завжди можлива і веде до ейлерового ланцюга у графі G: виходячи з довільної вершини $u \in V$, ідемо по ребрах графа G довільним чином згідно з такими правилами:*

(i) стираємо ребра, які пройдені, і стираємо ізольовані вершини, які при цьому виникають;

(ii) на кожному етапі йдемо по мосту лише тоді, коли немає інших можливостей.

Доведення. Насамперед покажемо, що вказана процедура може бути виконана на кожному етапі. Нехай у процесі роботи процедури було досягнуто вершини v, тоді якщо $u \neq v$, то підграф H, який залишився, зв'язний і має рівно дві вершини непарного степеня — це вершини u і v. За наслідком 34 граф H має напівейлеровий ланцюг P із v в u. Оскільки вилучення непарного ребра ланцюга P не порушує зв'язності графа H, то звідси випливає, що побудова, яка дається в теоремі, завжди можлива на кожному етапі. Якщо ж $u = 0$, то доведення залишається таким самим доти, поки ще є ребра, що інцидентні вершині u.

Залишається лише показати, що дана процедура завжди приводить до повного ейлерового ланцюга. Але це очевидно, оскільки в G не може бути ребер, які не були пройдені після використання останньо-

го ребра, яке інцедентне вершині u (інакше вилучення деякого ребра, суміжного з одним із тих, які залишилися, веде до незв'язного графа, а це суперечить умові (ii)).

5.4.7. Властивості гамільтонових графів

Розглянута вище проблема існування замкнутого ланцюга, який проходить через кожне ребро заданого зв'язного графа G, аналогічно може бути сформульована і для вершин. Тобто, чи існує замкнутий ланцюг у заданому зв'язному графі G, який проходить рівно один раз через кожну вершину графа G. Очевидно, що такий ланцюг повинен бути циклом. Якщо такий цикл у графі G існує, то він називається **гамільтоновим циклом**, а граф G — **гамільтоновим графом**.

Пошук критерія гамільтоновості графа — це одна з основних невирішених проблем теорії графів. Про гамільтонові графи відомо ще зовсім мало. Одним з найвідоміших результатів загального характеру є

Теорема 119 **(Дірак).** *Якщо у графі* $G = (V, E)$ *з* $n \geq 3$ *вершинами* $(\forall v \in V)\ n(v) \geq n/2$, *то граф* G *є гамільтоновим.*

Доведення. Введемо у граф G k нових вершин і з'єднаємо кожну з цих вершин з кожною вершиною графа G. Будемо вважати, що k — найменше число вершин, які потрібно ввести у граф G для того, щоб одержаний граф G' став гамільтоновим.

Припустимо, що $k > 0$ і $v, p, w,..., v$ — гамільтоновий цикл, де v і w — вершини з G, а p — одна з нових вершин. Тоді v і w не є суміжними, оскільки в протилежному випадку вершина p зайва, а це суперечить мінімальності k. Більше того, вершина (наприклад w'), суміжна з вершиною w, не може безпосередньо йти за вершиною v', яка суміжна з v, оскільки тоді можна було б замінити $v, p, w,..., v', w',..., v$ на $v, v',..., w, w',..., v$, перевернувши частину циклу, яка знаходиться між вершинами w і v'. Звідси випливає, що число вершин графа G', які не є суміжними з w, не менше ніж число вершин суміжних з v, тобто, у крайньому випадку, дорівнює $n/2 + k$. З іншого боку, очевидно, що число вершин графа G', які суміжні з w, теж не менше ніж $n/2 + k$. Оскільки довільна вершина графа G' не може бути одночасно суміжною і несуміжною з w, то загальна кількість вершин графа G', яка дорівнює $n + k$, не менша ніж $n + 2k$. Це може бути лише тоді, коли $k = 0$. Отримана суперечність доводить теорему. ■

Якщо у графі $G = (V, E)$ порядку n зафіксувати одну з вершин і обхід графа завжди починати з неї, то довільному гамільтоновому

циклу буде відповідати перестановка елементів множини V. Отже, знайти гамільтонів цикл або переконатися в його відсутності можна шляхом перебору $(n-1)!$ перестановок. Якщо граф G гамільтонів, то цей перебір у повному обсязі необхідно буде зробити лише у випадку великого невезіння, тобто коли перестановка, що відповідає гамільтоновому циклу, зустрічається в цьому процесі останньою. Якщо ж граф G — не гамільтонів, то діючи таким чином, необхідно перебрати всі $(n-1)!$ перестановок.

Хоча на практиці користуються різними алгоритмами часткового перебору, але складність цих алгоритмів залишається високою (пропорціональною $(n-1)!$).

5.5. Матриці і графи

5.5.1. Матриці суміжностей і досяжності

Нехай $G = (V, E)$ — скінченний граф порядку n, тоді йому можна поставити у відповідність квадратну матрицю $A(G)$ розмірності $n \times n$, яка має вигляд

$$a_{ij} = \begin{cases} 1, & \text{коли вершини } v_i \text{ і } u_j \text{ суміжні;} \\ 0, & \text{у протилежному випадку.} \end{cases}$$

Матриця $A(G)$ називається **матрицею суміжності** графа G. В силу симетричності відношення суміжності матриця суміжностей графа повинна бути симетричною, а число одиниць в i-му рядку дорівнювати степеня вершини u_i.

Задання графів за допомогою матриць суміжностей дозволяє сформулювати простий критерій ізоморфності графів.

Теорема 120. *Графи ізоморфні тоді і тільки тоді, коли їх матриці суміжностей можна отримати одну з другої однаковими перестановками рядків і стовпчиків.*

Доведення. Перенумеруємо вершини графів $G = (V, E)$ і $H = (V_1, E_1)$ ($|V| = |V_1| = n$) цілими числами від 1 до n.

Якщо $A(G) = B(H)$, то все доведено. У протилежному випадку графи G і H відрізняються лише нумерацією вершин. Отже, існує така підстановка s на множині вершин V, яка зберігає суміжність, тобто якщо $(u, v) \in E$, то $(s(u), s(v)) \in E_1$. Тоді маємо $b_{s(i)s(j)} = a_{ij}$ $i, j = 1, 2, \ldots, n$. ■

Доведена теорема свідчить, що представлення графа матрицею суміжностей виявляє важливу інформацію про природу графа. Пам'ятаючи про те, що коли $A(R)$ — матриця суміжностей відношення R, то $A(R)^2$ — матриця суміжностей відношення R^2, $A(R)^3$ — матриця суміжностей відношення R^3 і т. д., має місце таке твердження.

Теорема 121. *Якщо A — матриця суміжностей графа G, який має порядок n, то елемент b_{ij} матриці A^k є числом маршрутів довжини k, які з'єднують вершини u_i і u_j у графі G.*

Доведення індукцією за величиною числа k.

Для $k = 1$ твердження очевидне, оскільки маршрут довжини 1 є ребром графа G.

Нехай теорема справедлива для всіх $k < m$. Покажемо, що вона має місце і для $k = m$.

Нехай b_{iq} — елемент матриці A^{m-1}, a_{qj} — елемент матриці A. Тоді в силу припущення індукції, $b_{iq}a_{qj}$ — це число маршрутів довжини m, які з'єднують вершини u_i і u_j і проходять через вершину u_q. Отже, сума

$$\sum_{q=1}^{n} b_{iq}a_{qj} \qquad (*)$$

є числом маршрутів довжини m, які з'єднують вершини u_i і u_j у графі. ■

Наслідок 31. *У графі порядку n маршрут, який з'єднує вершини u і v, існує тоді і тільки тоді, коли відповідний цим вершинам елемент матриці*

$$C = \sum_{k=1}^{n} A^k$$

не дорівнює нулю.

Доведення. Якщо відповідний елемент $c \neq 0$, то в силу доведених раніше тверджень існує ланцюг, який з'єднує ці вершини u, u_1,..., v довжини не більше ніж n.

Якщо маршрут, який з'єднує u і v, існує, то існує маршрут довжини $k < n$, який з'єднує ці вершини. За теоремою 121 у матриці C відповідний елемент c повинен бути відмінним від нуля. Що й потрібно було довести. ■

Зауважимо, що в деяких випадках при розв'язуванні задач на графах знаходження матриці C і степенів матриці A можна значно спростити, якщо при обчисленні суми (*) використовувати булеві зв'язки $\vee$, $\&$, тобто обчислювати

$$C = \bigvee_{q=1}^{n} a_{iq} \,\&\, a_{qj}.$$

У результаті елементи матриць спрощуються і обчислення виконуються більш ефективно. Одержану таким чином матрицю C будемо називати **матрицею досяжності графа.**

Наслідок 32. *Граф зв'язний тоді і тільки тоді, коли кожний елемент матриці досяжності графа дорівнює одиниці.*

Аналогічно визначається матриця суміжності орграфа $G = (V, E)$

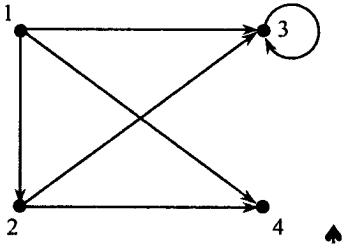
$$a_{ij} = \begin{cases} 1, \text{ якщо } (u_i, u_j) \in E; \\ 0, \text{ у протилежному випадку.} \end{cases}$$

Матриці суміжності неорієнтованого графа і орграфа відрізняються між собою тим, що перша завжди симетрична, а друга не обов'язково симетрична. Очевидно, що довільна квадратна матриця, елементи якої 0 і 1, буде матрицею суміжності деякого орієнтованого графа.

Приклад 5.5.10. Нехай маємо матрицю

$$\begin{pmatrix} 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

Цій матриці відповідає такий орграф:



♠

Із теореми про ізоморфізм графів випливає, що ранги матриць суміжності ізоморфних графів рівні між собою. Це дає можливість ввести таке визначення.

Рангом графа G називається ранг його матриці суміжності, який будемо позначати через $rank(G)$.

5.5.2. Матриця Кірхгофа

Нехай $G = (V, E)$ — граф порядку n і $V = \{1, 2,..., n\}$. Визначимо матрицю $B(G)$ для графа G таким чином

$$b_{ij} = \begin{cases} -1, & \text{якщо } (i,j) \in E \\ 0, & \text{якщо } i \neq j \text{ і } (i,j) \in E \\ n(i), & \text{якщо } i = j, \end{cases}$$

де $n(i)$ — степінь вершини i. Матриця $B(G)$ називається *матрицею Кірхгофа* графа G. Сума елементів кожного рядка і кожного стовпчика цієї матриці дорівнює нулю.

Зауважимо, що теорема про ізоморфізм графів залишається справедливою, якщо розглядати не матриці суміжності, а матриці Кірхгофа. Доведення повністю зберігається також.

Теорема 122. *Нехай B — довільна числова квадратна матриця розмірності n, у кожному рядку і стовпчику якої сума елементів дорівнює нулю, тобто*

$$\sum_{j=1}^{n} b_{ij} = 0 \ (i = 1,2,...,n) \ \sum_{i=1}^{n} b_{ij} = 0 \ (j = 1,2,...,n).$$

Тоді алгебраїчні доповнення всіх елементів матриці B рівні між собою. Зокрема, таку властивість має матриця Кірхгофа довільного графа.

Доведення. Очевидно, що $rank(B) < n$. Якщо $rank(B) < n - 1$, то алгебраїчні доповнення всіх елементів цієї матриці дорівнюють нулю. Нехай $rank(B) = n - 1$ і C — матриця, елементами якої є c_{ij}, що є алгебраїчними доповненнями елемента b_{ij} матриці B, $i = 1, 2,..., n, j = 1, 2,..., n$. Відомо, що $BC = |B|E$, де $|B|$ — визначник матриці B, а E — одинична матриця. Отже, $BC = 0$, оскільки $|B| = 0$ ($rank(B) = n - 1$). Для стовпчика матриці C з номером j ($j = 1, 2,..., n$) справедливі рівності

$$b_{i1}c_{1j} + b_{i2}c_{2j} + \ldots + b_{in}c_{nj} = 0,$$

$i = 1, 2,..., n$. Ці рівності можна розглядати як систему лінійних однорідних рівнянь з матрицею B відносно невідомих $b_{1j}, b_{2j}, \ldots, b_{nj}$. Оскільки $rank(B) = n - 1$, то всі розв'язки цієї системи пропорціональні. Крім того, вектор (1, 1,..., 1) задовольняє даній системі і тому $b_{1j} = b_{2j} = \ldots = b_{nj}, j = 1, 2,..., n$. Враховуючи те, що і $CB = 0$, то аналогічно отримуємо

$$b_{1j} = b_{2j} = \ldots = b_{nj}, j = 1, 2,..., n.$$

Отже, $b_{ij} = b_{kl}$; $i, j, k, l = 1, 2,..., n$. ■

5.5.3. Матриця інцидентності графа

Нехай $G = (V, E)$ — граф, де $V = \{1, 2,..., n\}$, $E = \{e_1,..., e_m\}$. Визначимо матрицю $I(G)$ такими умовами:

$$i_{kl} = \begin{cases} 1, & \text{якщо вершина } k \text{ і ребро } e_l \text{ інцидентні;} \\ 0, & \text{у протилежному випадку.} \end{cases}$$

Матриця $I(G)$ називається *матрицею інцидентності графа G.*

Для орієнтованих графів визначення матриці інцидентності $I(G)$ дещо змінюється

$$i_{kl} = \begin{cases} 1, & \text{якщо вершина } k \text{ — початок дуги } e_l \\ -1, & \text{якщо вершина } k \text{ — кінець дуги } e_l \\ 0, & \text{якщо вершина } k \text{ і дуга } e_l \text{ не інцидентні.} \end{cases}$$

Для матриць інцидентності має місце твердження, аналогічне теоремі про ізоморфізм графів.

Теорема 123. *Графи (орграфи) ізоморфні тоді і тільки тоді, коли їх матриці інцидентності можна одержати одну з другої довільними перестановками рядків і стовпців.*

Нехай G — граф. Цей граф можна перетворити в орграф, якщо кожному ребру надати одну з двох можливих орієнтацій. Одержаний таким чином орграф називається *орієнтацією графа G.* Безпосередньою перевіркою встановлюється справедливість такого твердження.

Теорема 124. *Якщо B — матриця Кірхгофа графа G, а I — матриця інцидентності деякої його орієнтації H (вершини в H перенумеровані так само, як і в G), то $B = II^t$, де I^t — матриця, транспонована до матриці I.*

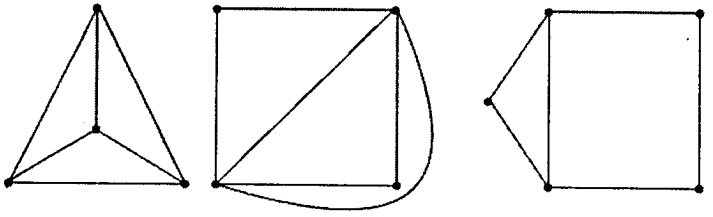
5.6. Планарність і укладання графів

Розглянемо питання про те, чи кожний граф можна зобразити на площині так, щоб його ребра не перетиналися.

5.6.1. Плоскі і планарні графи

Визначення 123. *Плоским називається граф, вершини якого є точками площини, а ребра — неперервними плоскими лініями без самоперетинів, які з'єднують відповідні вершини так, що ніякі два ребра не мають спільних точок, окрім інцидентної їм обом вершини.*

Прикладами плоских графів можуть служити такі графи:



Визначення 124. *Граф називається планарним, якщо він ізоморфний деякому плоскому графу.*

Граф K_n, зображений на рис. 5.6.11, планарний, оскільки він ізоморфний графам а) і б), зображеним вище.

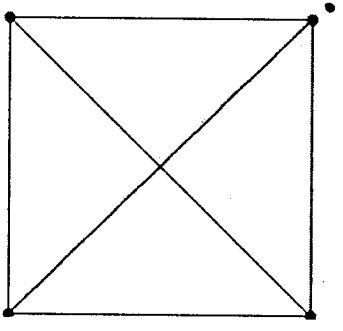


Рис. 5.6.11. Планарні графи

Про планарні графи говорять, що вони мають плоску укладку, або що вони укладаються на площині. Можна визначити укладку графів не тільки на площині, а й на інших поверхнях і в просторі. Для цього необхідно ввести поняття жорданової кривої.

Визначення 125. *Жордановою кривою на площині називається неперервна крива лінія без самоперетинів. Замкнутою жордановою кривою називається жорданова крива, початок і кінець якої збігаються.*

Аналогічно можна визначити жорданову криву в трьохвимірному просторі або на таких поверхнях, як сфера, тор тощо. Має місце знаменита теорема Жордана.

***Теорема 125* (Жордан).** *Якщо S — замкнута жорданова крива на площині, а x і y — дві різні точки цієї кривої, то довільна жорданова крива, яка з'єднує точки x і y, або повністю лежить усередині S, окрім точок x і y, або поза кривою S, окрім точок x і y, або перетинає криву S у деякій точці, відмінній від точок x і y* (*див. рис.* 5.6.12).

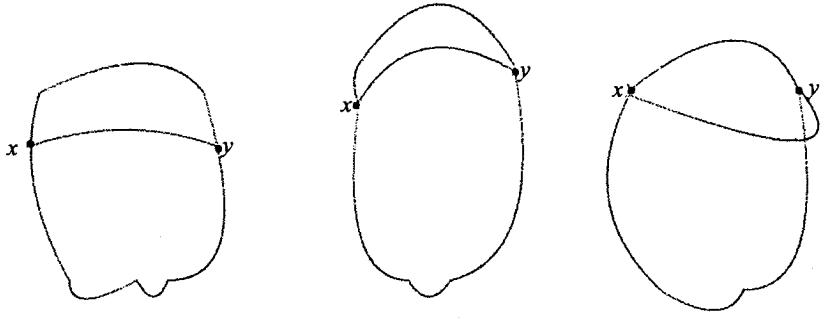


Рис. 5.6.12

Визначення 126. *Граф G укладається в просторі L, якщо існує взаємно однозначне відображення вершин і ребер графа G відповідно в точки і жорданові криві цього простору таке, що криві, які відповідають різним ребрам, перетинаються в інцидентних цим ребрам вершинах. Зображений таким чином граф G у просторі L називається* **укладкою графа** G.

Теорема 126. *Довільний граф укладається в трьохвимірному просторі.*

Доведення. Розмістимо всі вершини графа $G = (V, E)$ на осі OX. Із пучка площин, які проходять через цю вісь, виберено $|E|$ різних площин. Далі кожне ребро $(u, v) \in E$ зобразимо у відповідній площині півколом, яке проходить через вершини u і v. Зрозуміло, що в результаті отримаємо укладання графа G в трьохвимірний простір, оскільки всі ребра лежать у різних площинах і тому не перетинаються ні в яких точках, крім вершин.

Теорему доведено.

Теорема 127. *Граф укладається на сфері тоді і тільки тоді, коли він планарний.*

Доведення. Розглянемо стереографічну проекцію. Для цього проведемо площину Q, яка дотикається до сфери так, щоб точка N, діаметрально протилежна точці дотику (північний полюс N), не збігалася з вершиною графа і не лежала на його ребрі. Нехай граф G укладений

на сфері. Розглянемо граф G', отриманий у результаті стереографічної проекції графа G з точки N на площину Q. Оскільки відповідність між точками сфери, відмінними від N, і їх стереографічними проекціями взаємно однозначна, то граф G плоский і ізоморфний графу G'. Отже, граф G планарний.

Обернене твердження доводиться аналогічно з урахуванням взаємно однозначної відповідності, про яку йшлося вище.

Теорему доведено.

Слід зауважити, що визначення плоского і планарного графів, як і багато інших тверджень, зберігаються і для мультиграфів, і псевдографів.

Перш ніж перейти до пошуку критерію планарності графа, розглянемо добре відому задачу-головоломку про три будинки і три колодязі. Є три будинки 1, 2, 3 і три колодязі 4, 5, 6 (рис. 5.6.13). Кожний мешканець може користуватися кожним із трьох колодязів. Одного дня мешканці будинків вирішили прокласти доріжки до колодязів так, щоб виключити можливість зустрічі, тобто прокласти доріжки так, щоб вони не перетиналися. Виникає питання: чи можна прокласти доріжки так, як цього вимагає задача? Усі спроби прокласти дев'ять доріжок так, щоб вони не перетинались і з'єднували будинки з колодязями, закінчуються невдало. Виявляється, що це зовсім не випадково.

Наведена задача наводить на думку, що існують не тільки планарні графи. І це стверджується в наступній теоремі.

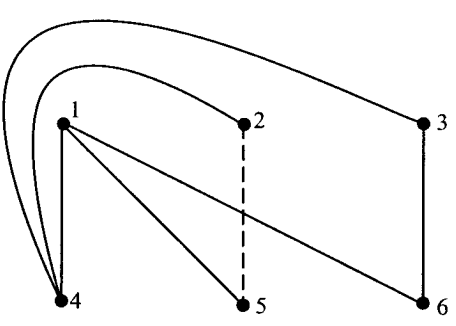


Рис. 5.6.13

Теорема 128. *Графи K_5 і $K_{3,3}$ не є планарними.*

Доведення. Припустимо супротивне, що граф K_5 планарний. Оскільки він має цикл довжини 5, наприклад v, w, x, y, z, то, без обмеження загальності, можна вважати, що за довільної укладки цього

графа на площині зазначений цикл зображується правильним п'ятикутником (рис. 5.6.14).

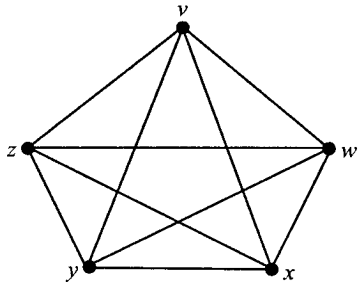


Рис. 5.6.14

За теоремою Жордана ребро (z, w) має лежати або цілком усередині цього п'ятикутника, або цілком поза ним. Третю можливість (коли ребро має спільну точку з п'ятикутником) не розглядаємо, оскільки маємо справу з укладкою на площині.

Розглянемо спочатку випадок, коли (z, w) лежить усередині п'ятикутника. Оскільки ребра (v, x) і (v, y) не повинні перетинати ребро (z, w), то обидва ці ребра лежать поза п'ятикутником: ця ситуація зображена на рис. 5.6.15.

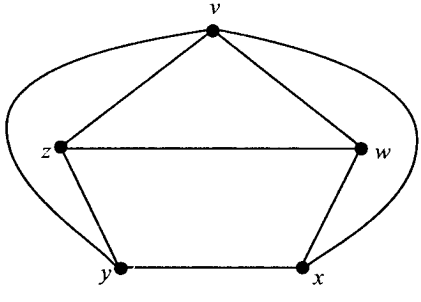


Рис. 5.6.15

Ребро (x, z) не може перетинати ребро (v, y) і тому воно має лежати всередині п'ятикутника. Аналогічно і ребро (w, y) теж має лежати всередині п'ятикутника. Але в цьому випадку ребра (w, y) і (x, z) обов'язково перетнуться і ми отримуємо суперечність з нашим припущенням.

Другий випадок доводиться цілком аналогічно до розглянутого і пропонується як вправа.

Розглянемо тепер граф $K_{3,3}$. Припустимо, що він планарний. Тоді існує цикл довжини 6, наприклад u, v, w, x, y, z, який можна зобразити у вигляді шестикутника (рис. 5.6.16). Виконуючи доведення так, як це робилося вище для K_5, маємо ситуацію, коли двоє з ребер (u, x), (v, y), (w, z) повинні лежати або всередині шестикутника або поза шестикутником, а тому вони не можуть не перетинатися. Отримана суперечність повністю доводить теорему.

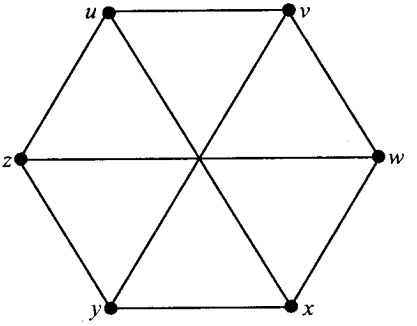


Рис. 5.6.16

Безпосередньо з визначення планарності графа випливають такі очевидні твердження.

Твердження 16. *Довільний підграф планарного графа теж планарний.*

Твердження 17. *Якщо граф містить непланарний підграф, то й сам граф непланарний.*

З останнього твердження випливає, що довільний граф, який включає K_5 і $K_{3,3}$ як підграфи, не буде планарним. Виявляється, що графи K_5 і K_{33}, по суті, єдині непланарні графи в тому плані, що кожний непланарний граф включає один із них як підграф. Для формулювання цього результату необхідно ввести поняття *гомеоморфні графи.*

Визначення 127. *Два графи називаються гомеоморфними або тотожними з точністю до вершин степеня* 2, *якщо вони можуть бути отримані з одного й того самого графа за допомогою операції введення вершини в ребро степеня* 2.

Наприклад, графи, зображені на рис. 5.6.17, гомеоморфні.

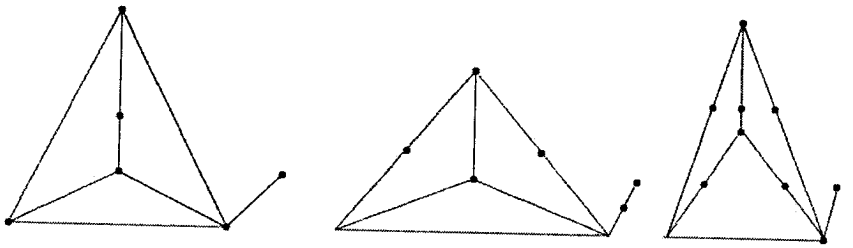


Рис. 5.6.17

Неважко переконатися, що гомеоморфізм графів є відношенням еквівалентності.

Теорема 129 **(Понтрягін-Куратовський).** *Граф планарний тоді і тільки тоді, коли він не має підграфів, гомеоморфних K_5 або $K_{3,3}$.*

Доведення цієї теореми не наводиться, оскільки воно досить складне і вимагає розгляду деяких понять, які виходять за рамки даного посібника. Для компенсації цього, виходячи з теореми Понтрягіна-Куратовського, розглянемо інший критерій планарності графів.

Теорема 130. *Граф планарний тоді і тільки тоді, коли він не має підграфів, які стягуються до графів K_5 і $K_{3,3}$.*

Доведення. Припустимо, що граф G не планарний. Тоді за теоремою Понтрягіна-Куратовського він має підграф H, гомеоморфний K_5 чи K_{33}. Стягуючи ті ребра із H, які інцидентні вершинам степеня 2, ми бачимо, що H стягується до K_5 або до $K_{3,3}$.

Нехай тепер граф G включає підграф H, який стягується до $K_{3,3}$ і вершина v графа $K_{3,3}$ отримана за допомогою операції стягування підграфа H_v графа H (рис. 5.16.8).

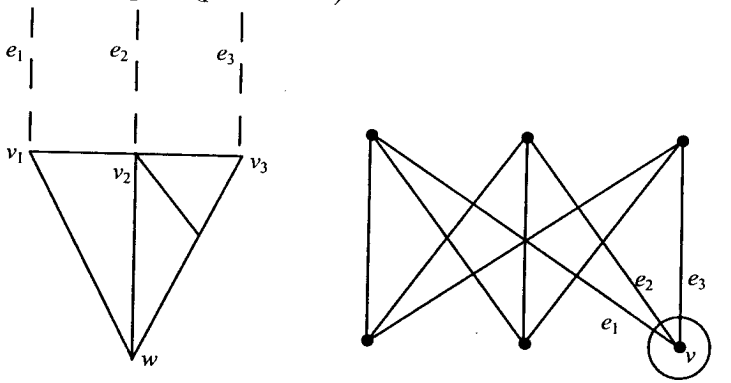


Рис. 5.6.18

У графі $K_{3,3}$ вершина v інцидентна трьом ребрам e_1, e_2 і e_3; як ребра із H вони інцидентні трьом (не обов'язково різним) вершинам v_1, v_2, v_3 підграфа H_v. Якщо v_1, v_2, v_3 різні, то можна знайти вершину w із H_v і три простих ланцюги, що ведуть із w в ці вершини і перетинаються тільки у вершині w. Можна виконати аналогічні побудови і в тому випадку, коли не всі вершини різні; при цьому прості ланцюги перетворюються в окремі вершини. Отже, підграф H_v можна замінити вершиною w з трьома простими ланцюгами, що виходять з цієї вершини. Якщо такі побудови виконати для кожної вершини графа $K_{3,3}$, а отримані прості ланцюги доповнити відповідними ребрами із $K_{3,3}$, то зрозуміло, що отриманий підграф буде гомеоморфним графу $K_{3,3}$. Згідно з теоремою Понтрягіна-Куратовського маємо, що граф G непланарний (рис. 5.6.19).

Аналогічно доводиться і випадок, коли G включає підграф, який стягується до графа K_5. ■

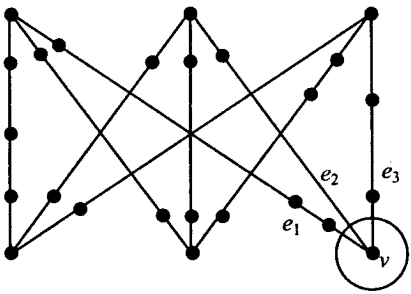


Рис. 5.6.19

5.6.2. Грані плоского графа. Формула Ейлера

У цьому підрозділі встановимо формулу Ейлера, яка зв'язує число вершин ребер і граней зв'язного плоского графа. Визначимо спочатку поняття грані плоского графа G.

Точка x площини S, на якій розміщений граф G, називається *диз'юнктною із* G, якщо точка x не відповідає жодній вершині графа G і не лежить на жодному ребрі цього графа. Дві точки площини $S - x$ і y — називаються еквівалентними, якщо вони диз'юнктні із G і їх можна з'єднати такою жордановою кривою, усі точки якої диз'юнктні із G (рис. 5.6.20). Введене відношення між точками площини є відношенням еквівалентності і, отже, множина всіх точок площини S розбивається на класи еквівалентності, які й називаються *гранями графа* G. Наприклад, граф G, зображений на рис. 5.6.21, має чотири грані f_1, f_2, f_3, f_4 і грань f_4 — нескінчена його грань.

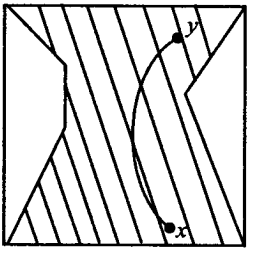


Рис. 5.6.20

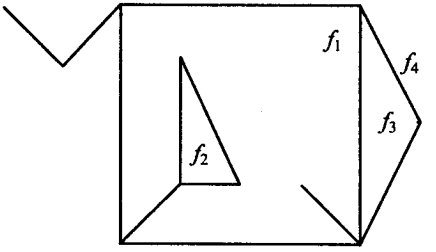


Рис. 5.6.21

***Теорема 131* (Ейлер).** *Нехай G — зв'язний плоский граф, у якому n, m і f відповідно число вершин, ребер і граней. Тоді має місце рівність $n + f = m + 2$.*

Доведення виконується індукцією за числом ребер у графі G.

Якщо $m = 0$, то $n = 1$ (оскільки G зв'язний) і $f = 1$ (нескінченна грань), тобто в цьому випадку теорема має місце.

Нехай теорема істинна для довільного графа G, який має $m - 1$ ребро. Додамо до G нове ребро e (зауважимо, що це не є операція введення ребра у граф), тоді можливі такі випадки:

a) ребро e є петлею, і в цьому випадку число граней збільшується на одну, а число вершин залишається незмінним;

b) ребро e з'єднує дві різні вершини в G, і в цьому випадку одна із граней графа G розпадається на дві, число граней зростає на одиницю, а число вершин залишається незмінним;

c) ребро e інцидентне лише одній з вершин графа G, і в цьому випадку число його вершин зростає на одиницю, а число граней залишається незмінним.

У кожному з цих випадків теорема залишається справедливою, а оскільки наведеними випадками вичерпуються всі можливості, то цю теорему доведено повністю. ■

Теорему Ейлера легко перенести і на незв'язні графи.

Наслідок 33. *Нехай G — плоский граф з n вершинами, m ребрами, f гранями і k компонентами зв'язності, тоді $n + f = m + k + 1$.*

Доведення. Результат випливає завдяки застосуванню теореми Ейлера окремо до кожного з компонентів. При цьому нескінченна грань рахується лише один раз.

Наслідок 34. *Якщо G — зв'язний планарний граф з m ребрами і $n \geq 3$ вершинами, то $m \leq 3n - 6$.*

Доведення. Оскільки кожна грань графа обмежена не менше ніж трьома ребрами, то при підрахунках кількості ребер навколо кожної з граней отримаємо, що $3f \le 2m$. Множник 2 з'являється тому, що кожне ребро обмежує не більше двох граней. Отже, маємо

$$3(m+2) = 3m + 6 = 3(n+f) \le 3n + 2m,$$

тобто $m \le 3n - 6$. ■

Наслідок 35. *Графи $K_{3,3}$ і K_5 не є планарними.*

Доведення. Якби K_5 був планарним, то за наслідком 34 мали б, що $10 \le 9$. А це очевидна суперечність. ■

Для непланарності графа $K_{3,3}$ достатньо зауважити, що кожна його грань обмежена щонайменше чотирма ребрами і, отже, $4f \le 2m$, або $2f \le 9$. Але цього не може бути, оскільки з теореми Ейлера випливає, що $f = 5$.

Наслідок 36. *Кожний планарний граф має вершину, степінь якої не більший п'яти.*

Доведення. Без обмеження загальності можна вважати, що граф плоский, зв'язний і має хоча б три вершини. Якщо степінь кожної з його вершин не менший 6-ти, то маємо $6n \le 2m$, або $3n \le m$. За наслідком 34 маємо, що $3n \le 3n - 6$ — очевидна суперечність. ■

5.7. Розфарбування графів. Хроматичне число

Задачі розфарбовки вершин чи ребер графа займають важливе місце у теорії графів. До задачі побудови розфарбовки графа зводиться цілий ряд практичних задач.

5.7.1. Правильне розфарбування

Нехай $G = (V, E)$ — скінченний граф, а k — деяке натуральне число.

Визначення 128. *Довільна функція вигляду $f: V \to N_k$, де $N_k = \{1, 2, ..., k\}$, називається* **вершинною k-розфарбуванням**, *або просто* **k-розфарбуванням**, *графа G. Розфарбування називається* правильним, *якщо* $(\forall (u, v) \in E)\ (f(u) \ne f(v))$.

Граф, для якого існує правильне k-розфарбування, називається розфарбованим графом.

При визначенні розфарбовки графа множину N_k можна замінити довільною k-елементною множиною.

Правильне розфарбування графа можна трактувати як розфарбування кожної його вершини в один із k кольорів таким чином, щоб суміжні вершини були розфарбовані в різні кольори. Оскільки функція f не обов'язково взаємно однозначна, то при k-розфарбуванні фактично може бути використано менше k кольорів. Отже, правильне k-розфарбування можна розглядати як розбиття множини вершин V графа G на класи $V_1 \cup V_2 \cup \ldots \cup V_l = V$, де $l \le k$, $V_i \neq \varnothing$, $i = 1, 2, \ldots, l$. Кожний клас V_i є незалежною множиною (тобто множиною вершин, які несуміжні у графі), а самі класи називаються *кольоровими класами*.

Визначення 129. *Мінімальне число k, при якому існує правильне k-розфарбування графа G, називається* **хроматичним числом** *цього графа і позначається $X_p(G)$. Якщо $X_p(G) = k$, то граф G називається k-хроматичним. Правильне k-розфарбування графа G при $k = X_p(G)$ називається* **мінімальним**.

Приклад 5.7.12. Розглянемо граф G, зображений на рис. 5.7.22, де показано одне із правильних k-розфарбувань.

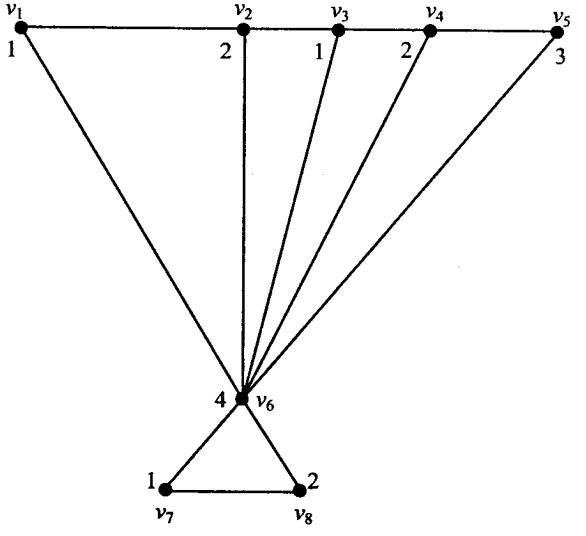


Рис. 5.7.22

Натуральними числами 1, 2, 3, 4 позначені фарби відповідних вершин. ♠

5.7.2. Практичні задачі, які зводяться до задачі розфарбування

Задача складання розкладу занять. Нехай потрібно прочитати декілька лекцій за найкоротший проміжок часу. Читання лекцій займає одну годину, але деякі лекції не можуть читатися одночасно (наприклад, їх читає один і той самий лектор). Побудуємо граф $G = (V, E)$, де V — множина, яка відповідає множині лекцій, причому дві вершини суміжні тоді і тільки тоді, коли відповідні лекції не можуть читатися одночасно. Очевидно, що довільне правильне розфарбування цього графа визначає допустимий розклад: лекції, які відповідають вершинам графа, що складають один кольоровий клас, читаються одночасно. Навпаки, довільний допустимий розклад визначає правильне розфарбування графа G. Оптимальний розклад відповідає мінімальним розфарбуванням, а число годин, необхідне для прочитування всіх лекцій, дорівнює $X_p(G)$.

Задача розподілу устаткування. Дано множини $V = \{v_1, v_2,..., v_n\}$ і $S = \{s_1, s_2, ..., s_m\}$ — види робіт і механізмів відповідно. Для виконання кожної з робіт необхідний певний час, однаковий для всіх робіт, і деякі механізми. При цьому жоден із механізмів не може одночасно використовуватися в декількох роботах. Необхідно розподілити механізми так, щоб загальний час виконання всіх робіт був мінімальним.

Побудуємо граф $G = (V, E)$, де $V = \{v_1,..., v_m\}$, а $(v_i, v_j) \in E$ тоді і тільки тоді, коли для виконання робіт v_i і v_j потрібен хоча б один спільний механізм. За правильного розфарбування графа G роботи, які відповідають вершинам одного кольору, можна використовувати одночасно, а найменший час виконання всіх робіт досягається при мінімальному розфарбуванні.

Задача проектування коробки швидкостей [13]. Коробка швидкостей — це механізм для зміни частоти оборотів валу, який регулює швидкість руху автомобіля. Ця зміна виконується за рахунок того, що шестерні, які знаходяться всередині коробки, вводяться в зачеплення спеціальним чином. Одна із задач, яка стоїть перед конструктором коробки, полягає в мінімізації її розмірів, а це часто зводиться до мінімізації числа валів, на яких розміщені шестерні.

Побудуємо граф $G = (V, E)$, де V — множина шестернів і $(u, v) \in E$, якщо шестерні u і v не можуть бути розміщені на одному валу (наприклад, вони повинні зчеплюватися, чи вони занадто важкі для одного валу). Вершини, які входять до одного кольорового класу, при правильному розфарбуванні цього графа, визначають шестерні, які можуть розміщуватися на одному валу, а хроматичне число $X_p(C)$ дорівнює мінімальній кількості валів, необхідних для коробки, що проектується.

5.7.3. Хроматичні числа деяких графів

Для деяких простих графів неважко знайти хроматичні числа. Наприклад,

$$X_p(K_n) = n, X_p(K_n - e) = n - 1, X_p(K_{m,n}) = 2,$$
$$X_p(C_{2n}) = 2, X_p(C_{2n+1}) = 3.$$

Очевидно, що граф є 1-хроматичним тоді і тільки тоді, коли він пустий, а 2-хроматичним — коли він двочастковий і непустий. 2-хроматичні графи, як правило, називають *біхроматичними*. Тому теорему Кьоніга про двочасткові графи можна сформулювати так.

Теорема 132. *Непустий граф біхроматичний тоді і тільки тоді, коли він не має циклів непарної довжини.*

За яких умов граф є 3-хроматичним невідомо, хоча легко знайти приклади таких графів. До них відноситься граф Петерсена, циклічні графи з непарним числом вершин, а також колеса з непарним числом вершин. Загалом про хроматичне число графа мало що можна сказати. Очевидно, що коли граф має n вершин, то його хроматичне число не перевищує n, а коли граф має підграф K_m, то його хроматичне число не менше за m. Значно більшу інформацію про хроматичне число графа можна отримати, якщо відомі степені кожної його вершини.

Нехай r означає максимальний степінь вершини графа $G = (V, E)$.

Теорема 133. *Для довільного графа G має місце нерівність* $X_p(G) \le r + 1$.

Доведення виконується індукцією за числом n вершин графа G.

Якщо $n \le r + 1$, то теорема очевидно має місце. Нехай теорема справедлива для всіх $n < k$. Покажемо її справедливість при $n = k$.

Якщо у графі G вилучити довільну вершину v, то в новому графі буде $n - 1$ вершина і, як і раніше, степінь кожної вершини цього графа не більший за r. За припущенням індукції цей граф розфарбовується $r + 1$ кольорами. Але звідси легко знайти розфарбування для графа G, якщо пофарбувати вершину v кольором, відмінним від кольорів, суміжних з вершиною v вершин. Оскільки цих вершин не більше r, то G розфарбовується $r + 1$ фарбою. ■

Наслідок 37. *Довільний кубічний граф розфарбовується за допомогою чотирьох фарб.*

Більш загальний результат, ніж попередня теорема, дає

Теорема 134 (*Брукс*). *Якщо G — зв'язний неповний граф і $r \geq 3$, то $X_p(G) \leq r$.*

Доведення цієї теореми опускається (його можна знайти, наприклад, у [39, 13]).

Хоча обидві ці теореми і дають певну інформацію про хроматичне число графа, але їх оцінки досить неточні. Дійсно, зірковий граф $K_{1,n}$, який за теоремою Брукса розфарбовується n фарбами, насправді є біхроматичним. Ця ситуація значно спрощується, якщо обмежитися планарними графами. У цьому випадку легко довести такий досить загальний і вагомий факт.

Теорема 135. *Для довільного планарного графа G має місце нерівність $X_p(G) \leq 6$.*

Доведення виконується індукцією за числом вершин графа G і пропонується як корисна вправа.

Більш детальний аналіз шляхів зниження верхньої межі хроматичного числа приводить до так званої *теореми про п'ять фарб.*

Теорема 136. *Для довільного планарного графа G має місце нерівність $X_p(G) \leq 5$.*

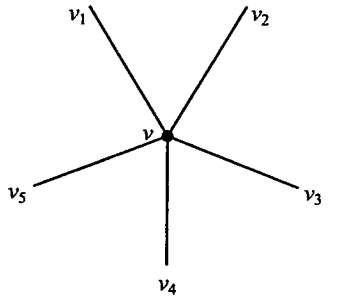
Доведення ведеться індукцією за числом n вершин графа G. Для планарних графів порядку менше шести результат очевидний.

Нехай G — планарний граф з n вершинами, і для всіх планарних графів з $n - 1$ вершинами теорема має місце. Можна вважати, що G — плоский граф і що він має за наслідком 36 вершину v, степінь якої не більше 5. Як і раніше, вилучення вершини v з графа G приводить до графа з $n - 1$ вершинами, який у силу індуктивного припущення можна розфарбувати п'ятьма фарбами. Отже, наша задача звелася до того, щоб розфарбувати v одним із п'яти кольорів і на цьому закінчити доведення.

Якщо $n(v) < 5$, то v можна пофарбувати довільною фарбою, яка не використовується у фарбуванні суміжних вершин.

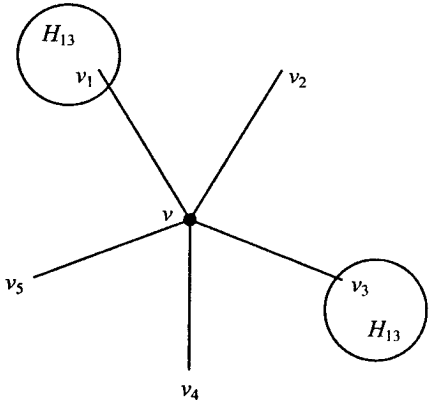
Розглянемо випадок $n(v) = 5$. Нехай суміжні вершини $v_1, v_2, \ldots, v_5$ розміщені навколо v за часовою стрілкою, так як це показано нижче на рисунку.

Якщо будь-яким двом вершинам v_i і v_j приписаний один і той самий колір, то цим усе доведено, оскільки v можна пофарбувати в колір, що не використовувався ні для якої з вершин v_i.



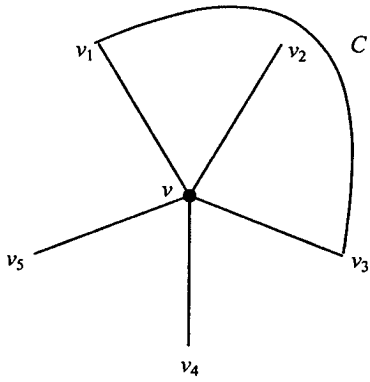
У результаті приходимо до останнього випадку, коли всім вершинам v_i приписані різні кольори. Нехай v_i пофарбована в колір c_i ($1 \le i \le 5$). Визначимо граф H_{ij} як підграф графа G, вершинами якого є всі вершини кольору c_i, а ребрами — усі ребра, що з'єднують вершину кольору c_i з вершиною кольору c_j. Тепер може бути лише два випадки.

1. v_1 і v_3 не належать одному й тому самому компоненту H_{13} (див. рис.)



У цьому випадку можна міняти колір усіх вершин компонента H_{13}, який не включає вершину v_1 (колір c_1 міняється на c_3, і навпаки). У результаті вершина v_1 отримує колір c_3, а це дозволяє вершину v пофарбувати кольором c_1 і закінчити доведення теореми для даного випадку.

2. v_1 і v_3 належать до одного компонента графа H_{13}. У цьому випадку існує цикл C вигляду $v, v_1, v_2,..., v_3, v$, частина якого, обмежена вершинами v_1 і v_3, цілком лежить усередині H_{13}.



Оскільки v_2 лежить усередині циклу C, а v_4 — зовні, то не існує простого ланцюга із v_2 у v_4, який цілком належить до H_{24}. Отже, можна поміняти кольори всіх вершин компонента підграфа H_{24}, якому належить вершина v_2. При цьому вершина v_2 набуває кольору v_4, що дає можливість пофарбувати вершину v кольором c_2. ■

Логічно поставити запитання: чи можна отримати більш сильний результат, ніж результат, наведений у теоремі 136? Це питання приводить нас до однієї з відомих невирішених проблем теорії графів — **гіпотези чотирьох фарб.**

5.7.4. Гіпотеза чотирьох фарб

Гіпотеза чотирьох фарб для графів формулюється так: **кожний планарний граф можна розфарбувати за допомогою чотирьох фарб.**

Вже більше століття математики намагаються розв'язати (довести або спростувати) дану проблему, але досі остаточного розв'язку так і не знайдено. Відомо, наприклад, що довільний планарний граф порядку менше 52 розфарбовується за допомогою чотирьох фарб.

Історично гіпотеза чотирьох фарб пов'язана з розфарбуванням географічних карт. Якщо в нас є географічна карта декількох країн світу, то цікаво знати, скільки необхідно мати фарб для такого розфарбування території цих країн, щоб ніякі дві сусідні країни не були пофарбовані одним і тим самим кольором. Можливо, найбільш звична форма гіпотези чотирьох фарб така: довільну карту можна розфарбувати за допомогою чотирьох фарб.

Щоб зробити це твердження точним, необхідно надати точне формулювання поняття «карта». Оскільки в розглянутій нами задачі про

розфарбування географічних карт потрібно, щоб країни, розміщені по обидва боки ребра, були розфарбовані різними кольорами, то необхідно, щоб вони не мали мостів. Таким чином, **карта** — зв'язний плоский граф, який не має мостів. Будемо говорити, що карта розфарбовується k фарбами, якщо її грані, що мають спільне ребро, можна розфарбувати k кольорами так, щоб ніякі дві суміжні грані не були одного кольору.

Гіпотеза чотирьох фарб для карт формулюється так: **кожна карта розфарбовується за допомогою чотирьох фарб.**

Має місце твердження, яке наводиться без доведення.

Теорема 137. *Гіпотеза чотирьох фарб для карт еквівалентна гіпотезі чотирьох фарб для планарних графів.*

Як видно з описаних вище результатів, задачі визначення хроматичного числа графа і побудови мінімального розфарбування довільного графа досить складні, а ефективні алгоритми їх розв'язку невідомі. Розглянемо простий алгоритм побудови правильного розфарбування, який у деяких випадках дає розфарбування, близькі до мінімальних.

РОЗФАРБ (V, E).

початок.

Довільній вершині v_1 із V приписати колір 1;

Якщо вершини v_1, v_2,..., v_i розфарбовані l кольорами 1, 2,..., l, де $l \le i$, то новій довільно вибраній вершині u_{i+1} приписати мінімальний колір, який не використовується у розфарбуванні вершин із множини См(v_{i+1}).

кінець.

На закінчення цього підрозділу зауважимо, що розфарбування, яке будує даний алгоритм, називається *послідовним*. Очевидно, що це розфарбування правильне. Для повних k-часткових графів послідовне розфарбування є мінімальним. У загальному випадку це не так.

5.8. Нескінченні графи

Розглянемо деякі поняття теорії нескінченних графів. Ці поняття здебільшого є узагальненнями визначень, які були розглянуті вище для скінченних графів.

Визначення 130. **Нескінченним (неорієнтованим) графом** *називається пара $G = (V, E)$, де V — нескінченна множина вершин і E — нескінченна множина ребер. Якщо обидві множини V і E — зліченні, то граф називається* **зліченним.**

Слід зазначити, що ці визначення виключають той випадок, коли V скінченна множина, а E нескінченна множина і коли E — скінченна множина, а V — нескінченна множина. У цих випадках граф G можна вважати скінченним графом, у якого в першому випадку скінченне число вершин і нескінченне число кратних ребер (або петель), а в другому випадку — скінченне число ребер і нескінченне число ізольованих вершин.

Визначення таких понять, як *суміжні вершини*, *інцидентні ребра*, *ізоморфні графи*, *підграф графа*, *зв'язний граф*, *компонент зв'язності*, повністю переносяться на нескінченні графи.

Степенем вершини v нескінченного графа називається потужність множини ребер, які інцидентні вершині v. Отже степінь вершини в нескінченному графі може бути як скінченним, так і нескінченним.

Нескінченний граф називається **локально скінченним,** якщо степінь кожної його вершини скінченний. Прикладом локально скінченного графа може бути квадратна решітка, складена з цілих чисел (рис. 5.8.23).

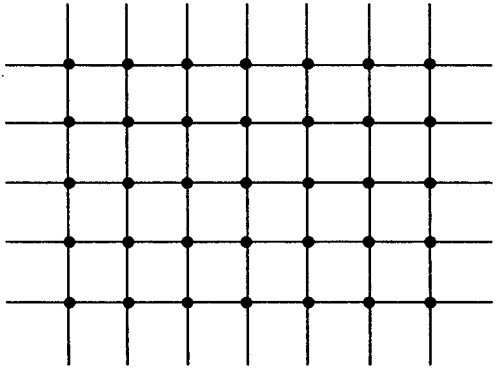


Рис. 5.8.23

Аналогічно можна визначити локально зліченний нескінченний граф: нескінченний граф називається **локально зліченним**, якщо кожна вершина цього графа має зліченний степінь.

Теорема 138. *Довільний зв'язний локально зліченний нескінченний граф є зліченним.*

Доведення. Нехай v — довільна вершина такого нескінченного графа і нехай A — множина всіх вершин, які суміжні з вершиною v ($A = \text{См}(u)$), B — множина всіх вершин, які суміжні з вершинами із

множини A і т. д. За умовою теореми множина A зліченна, множина B зліченна і т. д. В силу теореми 19 маємо, що об'єднання цих множин є множиною зліченною. Оскільки послідовність $\{v\}$, A, B,... в силу зв'язності графа включає кожну вершину нескінченного графа, то цим теорема доведена повністю.

Наслідок 38. *Довільний зв'язний локально скінченний нескінченний граф є зліченним.*

Доведення випливає з того, що об'єднання зліченної множини скінченних множин є зліченною множиною.

На нескінченні графи також переноситься і поняття маршруту, причому це поняття може мати такі різновиди:

1) *скінченний маршрут* у нескінченному графі G визначається так само, як і у випадку скінченних графів;

2) *нескінченним в один бік маршрутом* у графі G, який починається у вершині v_0, називається нескінченна послідовність ребер вигляду (v_0, v_1), (v_1, v_2), (v_2, v_3),...;

3) *нескінченним в обидва боки маршрутом у графі* G називається нескінченна послідовність ребер вигляду..., (v_{-2}, v_{-1}), $(v_{-1}; v_0)$, (v_0, v_1), (v_1, v_2),... .

Нескінченні в один і обидва боки ланцюги і прості ланцюги визначаються так само, як і поняття довжини ланцюга і відстані між вершинами. Наступна теорема, яка в теорії графів відома під назвою леми Кьоніга, надає умови існування наскінченних ланцюгів у графах.

Лема 6 **(Кьоніг).** *Нехай G — зв'язний локально скінченний нескінченний граф, тоді для довільної його вершини v існує нескінченний в один бік простий ланцюг, який починається у вершині v.*

Доведення. Якщо u — довільна вершина графа G, відмінна від вершини v, то існує деякий простий ланцюг, який з'єднує вершини v і u. Звідси випливає, що в G має бути нескінченно багато простих ланцюгів, з початком у вершині v. Оскільки степінь вершини v скінченний, то нескінченна множина таких простих шляхів має починатися з одного й того самого ребра. Якщо таким ребром є ребро (u, v_1), то, повторивши цю процедуру для вершини v_1, отримаємо нову вершину v_2 і відповідне їй ребро (v_1, v_2). Продовжуючи цей процес таким чином далі, отримаємо нескінченний в один бік простий ланцюг v, v_1, v_2, ... ■

Важливе значення леми Кьоніга полягає в тому, що вона дає можливість отримувати результати про нескінченні графи з відповідних результатів про скінченні графи.

Розглянемо на закінчення даного розділу нескінченні ейлерові та гамільтонові графи.

Визначення 131. *Зв'язний зліченний граф G називається* **ейлеровим**, *якщо в ньому існує нескінченний в обидва боки ланцюг, який включає кожне ребро графа G. Такий ланцюг називається (двостороннім)* **ейлеровим ланцюгом.**
Зліченний граф G називається **напівейлеровим**, *якщо в ньому існує нескінченний в один або в обидва боки ланцюг, який включає в себе кожне ребро графа G.*

Теорема 139. *Нехай G — зв'язний зліченний ейлерів граф, тоді:*
(а) у G кожна вершина має або нескінченний степінь, або парний степінь;
(б) для довільного скінченного підграфа H графа G доповнення H' графа H у графі G має не більше двох компонентів зв'язності;
(в) якщо H — скінченний підграф графа G і кожна вершина H має парний степінь, то доповнення H має рівно один нескінченний компонент зв'язності.

Доведення. (а) Нехай P — ейлерів ланцюг графа G, тоді при кожному проходженні по ланцюгу P через довільну вершину графа G степінь цієї вершини збільшується на два. Оскільки кожне ребро трапляється в P тільки один раз, то кожна вершина графа G має або нескінченний (зліченний) степінь, або цей степінь має бути парним.

б) Нехай P — ейлерів ланцюг графа G. Розіб'ємо P на три підланцюги P_-, P_0 і P_+ так, щоб P_0 являв собою скінченний ланцюг, який включає всі ребра графа H (і, можливо, ще деякі ребра графа G, що не належить графу H), а P_- і P_+ — два нескінченних в один бік ланцюги. Тоді нескінченний граф K, вершинами якого є вершини P_+ і P_-, а ребрами — ребра цих ланцюгів, має не більше двох компонентів зв'язності. Оскільки H можна одержати із графа K введенням лише скінченного числа ребер, то звідси випливає, що і H теж може мати не більше двох компонентів зв'язності.

в) Як і в попередньому випадку, розіб'ємо ейлерів ланцюг P на три підланцюги P_-, P_0 і P_+. Нехай u — початкова, v — кінцева вершина ланцюга P_0. Покажемо, що u і v з'єднані маршрутом у H.

Якщо $u = v$, то все зрозуміло, а якщо $u \neq v$, то вилучивши ребро (u, v) графа H, отримаємо, що вершини u і v мають непарні степені в H. А звідси, за наслідком 34, випливає, що u і v зв'язані напівейлеровим ланцюгом. Отже, H має рівно один компонент зв'язності. ■

5.9. Контрольні питання, задачі і вправи

Контрольні питання

1) Дайте визначення:
   а) ейлерового графа, ейлерового циклу;
   б) гамільтонового графа, гамільтонового циклу,
   в) хроматичного числа;
   г) розрізу в графі, мосту.
2) Який граф називається а) біхроматичним, б) k-хроматичним?
3) Які Ви знаєте властивості
   а) зв'язних, б) ейлерових, в) гамільтонових графів?
4) Який граф називається планарним?
5) Сформулюйте критерій планарності графа.
6) Чи можна довільний граф розфарбувати 4, 5, 6 фарбами?

Задачі і вправи

1. Довести, що кожне дерево є двочастковим графом.
2. Які дерева є повними двочастковими графами?
3. Побудувати остовні дерева для графів:
   а) K_5, б) $K_{3,3}$, в) графа Петерсена, г) платонових графів.
4. Знайти циклічні ранги графів:
   - K_n, $K_{m,n}$, N_n, W_n, платонових графів, графа Петерсена;
   - зв'язного регулярного графа степеня k і порядку n.
5. Побудувати остовні дерева і асоційовані з ними фундаментальні системи циклів і розрізів графів: K_6, $K_{3,4}$, W_5, C_6 і платонових графів.
6. Побудувати всі не ізольовані між собою дерева шостого порядку.
7. Нехай V — множина додатних цілих чисел від 1 до 20 і $a|b$ — відношення подільності, тобто $a|b$ означає, що число a є дільником числа b. Намалювати граф відношення $a|b$ для множини V.
8. Побудувати матриці суміжності та інцидентності для платонових графів.
9. Які з платонових графів мають ейлерові цикли; гамільтонові цикли?
10. Знайти ейлерів ланцюг для графа, зображеного на рис. 5.6.13.
11. Знайти хроматичні числа:
    а) платонових графів;
    б) об'єднання двох графів з хроматичними числами l і l' відповідно;
    в) з'єднання двох графів із хроматичними числами l і l' відповідно.

12. Нехай G — регулярний граф порядку n і степеня d. Показати, що $X_p(G) \geq n/(n-d)$

13. Граф називається **критичним**, якщо вилучення довільної його вершини приводить до графа з меншим хроматичним числом. Довести, що:

а) K_n є критичним для всіх $n > 1$;

б) C_n критичний тоді і тільки тоді, коли n непарне;

в) якщо n непарне, то з'єднання C_n і C_n є критичним графом, хроматичне число якого дорівнює 6.

14. Показати, що довільний k-хроматичний критичний граф $G=(V, E)$ має такі властивості:

а) G — зв'язний;

б) $(\forall v \in V)\, n(v) \geq k-1$;

в) не існує вершини v із V, вилучення якої веде до незв'язного графа.

15. Перевірити теорему Ейлера для платонових графів, W_n, $K_{2,n}$, і графа, яким є шахівниця.

5.10. Дерева

Перейдемо до розгляду одного з найпоширеніших серед застосувань різновидів графів — дерев.

5.10.1. Визначення дерева. Властивості дерев

Визначення 132. *Зв'язний ациклічний граф називається (неорієнтованим)* **деревом.** *Дерево називається* **кореневим**, *якщо в ньому виділена вершина, яка називається* **коренем**. **Остовним деревом графа** *G називається остовний підграф графа G, який є деревом.*

Безпосередньо з визначення дерева випливає

Теорема 140. *Граф є деревом тоді і тільки тоді, коли довільні дві його вершини зв'язані єдиним ланцюгом.*

Доведення. Нехай граф є деревом. Якщо допустити існування більше ніж одного ланцюга, який з'єднує довільні дві його вершини, то в такому графі існує цикл, тобто граф не може бути деревом.

Навпаки, оскільки довільні дві вершини графа з'єднані ланцюгом, то граф зв'язний, а в силу того, що цей ланцюг єдиний, він не має циклів. Отже, граф є деревом. ■

Наслідок 39. *Якщо T — дерево і u — його кінцева вершина, то граф $T - u$ теж є деревом.*

Дійсно, граф $T - u$ є підграфом дерева T, для якого виконуються всі умови попередньої теореми. ■

Наслідок 40. *Довільне непусте дерево має щонайменше дві кінцеві вершини і одне кінцеве ребро.*

Доведення виконується індукцією за числом вершин дерева з наступним застосуванням леми про рукостискання. ■

Ребро зв'язного графа називається *суттєвим*, якщо його вилучення веде до порушення зв'язності цього графа.

Наслідок 41. *У дереві кожне ребро є суттєвим.*

Доведення випливає з того, що коли вилучити ребро (u, v) в дереві T, то в силу єдності ланцюга, який з'єднує вершини u і v, будемо мати два компоненти зв'язності, один з яких включає вершину u, а другий вершину v. Отже, граф T не є зв'язним. ■

Теорема 141. *Якщо $T = (V, E)$ — дерево і вершина $v \notin V$, то граф $T' = (V \cup \{v\}, E \cup \{(u, v)\})$, де u — довільна вершина із V, теж є деревом.*

Доведення. Оскільки T — дерево, то існує єдиний ланцюг, який з'єднує довільну вершину u' з вершиною u. Оскільки вершина $v \notin V$, то введення одного кінцевого ребра (u, v) приводить до того, що з кожної вершини u' існує єдиний ланцюг, який з'єднує вершини u і v (оскільки такий ланцюг єдиний, який з'єднує вершини u' і u). Згідно з теоремою 140 граф T' є деревом. ■

Перелічені вище властивості дерев підсумовує така теорема.

Теорема 142. *Нехай граф T має n вершин. Тоді перелічені нижче твердження еквівалентні:*

1) T є деревом;

2) T не має циклів і має $n - 1$ ребер;

3) T — зв'язний граф і має $n - 1$ ребер;

4) T — зв'язний і кожне його ребро є мостом;

5) довільні дві вершини графа T з'єднані рівно одним простим ланцюгом;

6) T не має циклів, але введення нового ребра в T веде до виникнення рівно одного циклу.

Доведення. Випадок $n = 1$ очевидний. Нехай $n \geq 2$.

(1) $\rightarrow$ (2). Індукція за числом n. Для $n = 2$ твердження очевидне. Оскільки T не має циклів, то вилучення довільного ребра із T (згідно з наслідком 45) веде до розбиття T на два підграфи, кожний з яких є деревом. В силу індуктивного припущення в кожному з цих підграфів ребер на одиницю менше, ніж вершин. Отже, число всіх ребер у T дорівнює $n_1 - 1 + n_2 - 1 + 1 = n - 1$.

(2) $\rightarrow$ (3). Якщо T не зв'язний, то кожен його компонент зв'язності T_i є зв'язний граф без циклів. З (2) випливає, що кожна компонента зв'язності T_i має $n_i - 1$ ребер. Але тоді граф T має не більше ніж $n - 2$ ребер, що суперечить умові. Отже, граф T зв'язний і має $n - 1$ ребер.

(3) $\rightarrow$ (4). Вилучення довільнго ребра в T веде до графа, який має n вершин і $n - 2$ ребер. Але такий граф за наслідком 45 не може бути зв'язним.

(4) $\rightarrow$ (5). Оскільки T — зв'язний, то кожна пара його вершин з'єднана хоча б одним простим ланцюгом. Якщо ж деяка пара вершин з'єднується двома простими ланцюгами, то вони складають цикл, а це суперечить тому, що кожне ребро в T є мостом.

(5) $\rightarrow$ (6). Якщо T має цикл, то довільні дві вершини цього циклу з'єднані не менш ніж двома простими ланцюгами. Добавляючи до графа T деяке ребро e, ми отримуємо цикл, оскільки вершини, котрі інцидентні ребру e, уже з'єднані в T простим ланцюгом. Покажемо, що цей цикл єдиний. Дійсно, нехай вершини u і v несуміжні. За умовою u і v з'єднані єдиним простим ланцюгом. Якщо ввести ребро (u, v) в T, то простий ланцюг переходить у простий цикл. Якби існував при цьому ще деякий цикл, який включає ребро (u, v), то це суперечить тому, що u і v з'єднані єдиним простим ланцюгом. Якщо u і v суміжні, то теорема справедлива (з'являються кратні ребра, а це цикл).

(6) $\rightarrow$ (1). Нехай T незв'язний граф, тоді введення довільного ребра, яке з'єднує вершини з різних компонентів зв'язності, не приводить до появи циклу. А це суперечить умові (6). Отже, граф T не має циклів і зв'язний, тобто T — дерево. ■

Наслідок 42. *Нехай G — ліс з n вершинами і k компонентами, тоді G має $n - k$ ребер.*

Доведення. Із умови (2) попередньої теореми випливає, що кожен компонент G_i має $n_i - 1$ ребро. Але тоді число ребер у G дорівнює

$$(n_1 - 1) + (n_2 - 1) + \ldots + (n_k - 1) = n_1 + n_2 + \ldots + n_k - k = n - k. \blacksquare$$

Часто виникає питання, скільки можна побудувати різних дерев порядку n. Відповідь на це питання дає

***Теорема 143* (Келі).** *Число різних дерев, які можна побудувати на n вершинах, дорівнює* n^{n-2}.

Доведення. Нехай $V = \{v_1, v_2,..., v_n\}$ і $T = (V, E)$ — дерево. За наслідком 44 в T повинні бути кінцеві вершини. Нехай v_1 — перша з них і (u_1, v_1) — відповідне їй кінцеве ребро. Вилучимо з T це ребро і вершину v_1, позначимо у множині V вершину u_1 і занесемо її в список вершин L, який спочатку є пустим. За наслідком 43, отриманий граф знову є деревом. Застосуємо до нього таку саму процедуру, отримаємо нове дерево і список $L = [u_1, u_2]$ і т. д. Цю процедуру застосовуємо доти, поки після вилучення ребра (u_{n-2}, v_{n-2}) не залишиться єдине ребро (u_{n-1}, v_{n-1}). Тоді список $L = [u_1, u_2,..., u_{n-2}]$ однозначно визначається деревом T і двом різним деревам T і T' з n вершинами відповідають різні списки. Кожна вершина u з'являється в списку L $n(u) - 1$ раз.

Навпаки, кожний список L визначає дерево T за допомогою зворотної побудови. Якщо задано список L, то знаходимо першу позначену вершину v_1 у множині V, таку що $v_1 \in L$. Ця вершина визначає ребро (u_1, v_1). Далі стираємо позначку вершини u_1 у множині V, вилучаємо u_1 із L і продовжуємо побудову для нового списку L, який складається з $n - 3$ елементів. Отриманий у результаті такої побудови граф є деревом згідно з теоремою 140.

Оскільки різних елементів у списку L може бути не більше n^{n-2}, то цим і доводиться справедливість теореми. ■

Теорема Келі має багато різних інтерпретацій. Зупинимося на одній з них, яка відома під назвою **задачі про з'єднання міст**: необхідно з'єднати n міст залізницями так, щоб не будувати зайвих ліній; скількома способами можна побудувати таку систему доріг?

Практичне значення цієї задачі полягає ось у чому: нехай відома вартість $c(a, b)$ будівництва залізниці між містами a і b, яка мережа доріг, що з'єднує всі n міст, має найменшу можливу вартість?

Теорема Келі дає розв'язок цієї задачі. Для цього кожному дереву T, побудованому на n вершинах, співставляється сума

$$C(T) = \sum_{(a,b)\in E} c(a,b).$$

Вибравши серед цих сум найменшу, отримаємо розв'язок задачі. Для цього, як випливає з теореми Келі, необхідно переглянути не більше ніж n^{n-2} сум.

Як відомо з попереднього, вилучення у зв'язному графі G одного ребра, що належить деякому циклу, не порушує зв'язності графа G (див. теорему 109). Застосуємо цю процедуру до одного з

циклів, які залишилися у графі G, і так до тих пір, поки в G не залишиться жодного циклу. У результаті отримаємо дерево, яке включає всі вершини графа G. Це дерево називається **остовним деревом графа** G.

У загальному випадку нехай G є граф з n вершинами, m ребрами і k компонентами. Застосовуючи наведену вище процедуру до кожного компонента графа G, отримаємо в результаті граф, який називається **остовним лісом.** Число ребер, які при цьому вилучаються, називається **цикломатичним числом** або **циклічним рангом графа** G і позначається $C(G)$. Таким чином, цикломатичне число є мірою зв'язності графа.

Слід зауважити, що циклічний ранг дерева дорівнює нулю, а циклічний ранг циклічного графа — одиниці.

Нехай T — остовний ліс графа G. Граф G', отриманий з графа G шляхом вилучення всіх ребер графа T, називається *доповненням остовного лісу* T графа G.

Теорема 144. *Якщо T — остовний ліс графа G, то:*

а) кожний розріз у G має спільне ребро з T;

б) кожний цикл у G має спільне ребро з доповненням T.

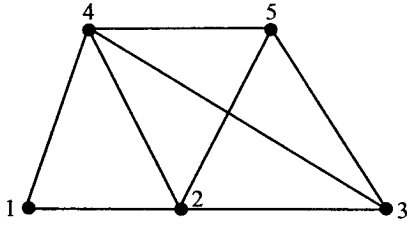
Доведення. а) Нехай M — розріз графа G, вилучення якого розбиває один з компонентів G на два підграфи H і F. Оскільки T — остовний ліс, то в ньому має існувати ребро, яке з'єднує вершину із H з вершиною із F. Але це ребро є шуканим ребром.

б) Якщо C — цикл у графі G, який не має жодного спільного ребра з доповненням T, то C входить до остовного лісу, а це неможливо, оскільки T — ліс. ■

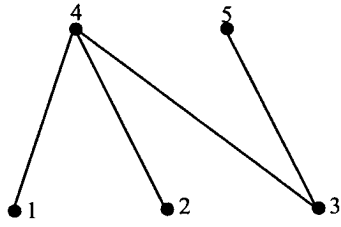
5.10.2. Фундаментальна система циклів графа

З поняттям остовного лісу T графа G тісно пов'язане поняття фундаментальної системи циклів, яка асоціюється з T. Нехай T — остовний ліс графа G. Якщо додати довільне ребро графа G, яке не входить у T, то за пунктом (6) теореми 142 дістанемо єдиний цикл. Множина всіх циклів, які отримані таким чином (тобто шляхом введення окремо кожного ребра у граф G, що не входить до T), називається **фундаментальною системою циклів**, яка асоційована з T. У тому випадку, коли нас не цікавить, який остовний ліс розглядається, будемо говорити про фундаментальну систему циклів графа G.

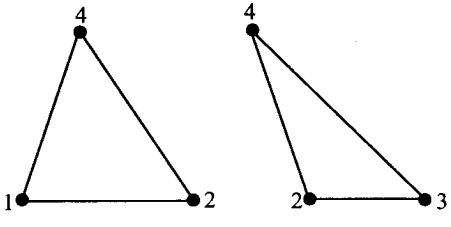
Приклад

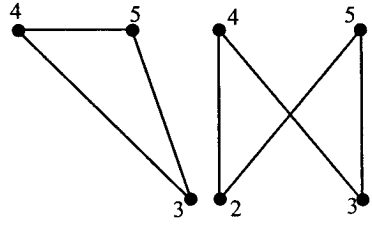


Граф G



Остовне дерево G





Фундаментальна система циклів, для остовного дерева графа G ♠

Теорема 145. *Нехай $G = (V, E)$ — довільний скінченний граф. Число ребер графа G, які необхідно вилучити для отримання остовного лісу T, не залежить від порядку їх вилучення і дорівнює $C(G) = |E| - |V| + k$, де k — число компонентів зв'язності графа G.*

Доведення. Нехай $G_i = (V_i, E_i)$, $(i = 1, 2,..., k)$ — один з компонентів зв'язності графа G. З теореми 142 (п. (2)) випливає, що для отримання остовного дерева T_i графа G_i необхідно вилучити $|E_i| - (|V_i| - 1)$ ребро. Тоді загальна кількість ребер, які необхідно вилучити у графі G, дорівнює

$$(|E_1| - (V_1 - 1)) + (|E_2| - (|V_2| - 1)) + \ldots + (|E_k| - (|V_k| - 1)) = \\ = |E| - |V| + 1. \blacksquare$$

Наслідок 43. *a) Граф G є лісом тоді і тільки тоді, коли $C(G) = 0$.*

b) Граф G має єдиний циклом тоді і тільки тоді, коли $C(G) = 1$.

c) Граф G, в якого число ребер перевищує число вершин, має цикл.

d) Довільне дерево порядку $n \geq 2$ має щонайменше дві кінцеві вершини.

Доведення. З леми про рукостискання і теореми 142 (п. (2)) випливає, що сума всіх степенів вершин дерева дорівнює $2(n - 1)$ і кожен степінь вершини більше нуля. Отже, хоча б два числа із усіх степенів вершин дерева дорівнюють одиниці. ■

Теорема 146. *Довільний ациклічний підграф довільного скінченного графа $G = (V, E)$ є підграфом остовного дерева графа G.*

Доведення. Нехай $H = (V', E')$ — ациклічний підграф графа G. Достатньо розглянути ситуацію, коли G зв'язний. Припустимо, що H незв'язний і H' — один з його компонентів зв'язності. Тоді в G існує таке ребро (u, v), що $u \in A$, а $v \in V' \setminus A$, де A — множина вершин графа H'. Тоді граф $H + (u, v)$ — ациклічний підграф графа G, який має менше компонентів зв'язності, ніж H. Повторюючи цю побудову далі, дійдемо до остовного дерева графа G, яке включає H як підграф.

Нехай тепер H зв'язний, але він не є остовним деревом графа G. Оскільки H — зв'язний ациклічний граф, то H — дерево, і оскільки H не є остовним деревом графа G, то в G існує вершина v, яка не належить H, і в G існує ребро (u, v), яке з'єднує вершину v з однією з вершин графа H, тоді граф $H + (u, v)$ є деревом, у якому H є підграфом. Якщо $H + (u, v)$ — остовне дерево G, то все доведено, а якщо ні, то повторюючи дану побудову, нарешті дійдемо до остовного дерева графа G, для якого H буде підграфом.

І якщо H — остовне дерево для G, то і в цьому випадку теорема має місце. ■

5.10.3. Остов найменшої ваги

Розглянемо таку задачу: нехай кожному ребру графа $G = (V, E)$ відповідає деяке число $d_i = d(e_i)$, $e_i \in E$, $i = 1, 2, ..., |E|$; необхідно у графі G знайти остовне дерево, сума чисел d_i в якому найменша. Число d_i у цьому випадку називається *вагою ребра* e_i, а сам граф G таким, що має *вагу*. Отже, задача полягає в тому, щоб знайти остовне дерево найменшої ваги. Ця задача виникає під час проектування доріг, ліній електропередач, трубопроводів і т. ін., коли необхідно з'єднати задані точки деякою системою зв'язку так, щоб довільні дві точки були зв'язані безпосередньо між собою, або через інші точки і щоб загальна довжина ліній зв'язку була мінімальною. Розглянемо методи розв'язку цієї задачі.

Перший розв'язок випливає безпосередньо з теореми Келі. Дійсно, розглянувши всі можливі остовні дерева повного графа G з n вершинами (а їх буде не більше ніж n^{n-2}), і вибравши те з них, яке має мінімальну вагову суму, ми розв'яжемо дану задачу. Але оскільки число n^{n-2} навіть при невеликому n буде великим, то такий шлях розв'язку даної задачі досить громіздкий. Розглянемо більш ефективні способи розв'язку цієї задачі. Один з таких способів дає алгоритм Краскала.

Алгоритм Краскала полягає у виконанні такої послідовності дій:

1) беремо пустий граф і будуємо граф $T_1 = O + e_1$, де e_1 — ребро графа $G = (V, E)$ мінімальної ваги.

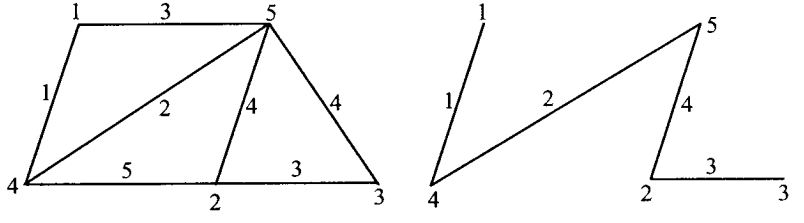
2) якщо граф T_k уже побудований і $k < n - 1$, то будуємо граф $T_{k+1} = T_k + e_{k+1}$, де e_{k+1} — ребро мінімальної ваги серед тих ребер графа G, які не ввійшли у граф T_k і яке не складає цикл з ребрами графа T_k.

Цей алгоритм ґрунтується на такій теоремі.

Теорема 147. *Нехай G — зв'язний граф порядку n і T — підграф графа G, отриманий у результаті виконання кроків* 1) *і* 2) *алгоритму Краскала. Тоді граф T — остовне дерево графа G мінімальної ваги.*

Доведення. Те, що граф T — остовне дерево графа G, випливає з теореми 142 (п. (2)). Залишається лише показати, що сума чисел d_i для всіх ребер T мінімальна. Припустимо супротивне, тобто що існує дерево S графа G таке, що його вага $P(S)$ менша за вагу дерева $T - P(T)$. Якщо e_i — перше ребро, яке не належить графу S, то підграф графа G, який дорівнює $S + e_i$, матиме єдиний цикл C, який включає ребро e_i. Оскільки C включає ребро e, яке належить S і не належить T, то підграф S', отриманий із S шляхом заміни e на e_i, також буде остовним деревом S графа G. За побудовою $d(e_i) \le d(e)$ і тому $P(S') \le P(S)$, причому число спільних ребер у S' і T на одне більше, ніж у S і T. Повторюючи крок за кроком цю процедуру, можна перетворити S на T так, щоб сума чисел d_i на кожному кроці не збільшувалась. Отже, $P(T) \le T(S)$ і ми отримали суперечність. ■

Приклад 5.10.13. Розглянемо граф, зображений на рис. 5.10.26, і знайдемо остовне дерево цього графа.



Граф G Остовне дерево мінімальної ваги для G

Рис. 5.10.26

5.11. Орієнтовані графи і дерева

5.11.1. Основні поняття

Нагадаємо, що коли ребра графа $G = (V, E)$ задані у вигляді упорядкованих пар (u, v), тобто $E \subseteq V^2$, *то* граф G називається **орієнтованим (орграфом)**. Ребра орграфа називаються **дугами**, вершина u — **початком**, вершина v — **кінцем дуги**. Дуга (u, v) називається дугою, що виходить з вершини u і веде у вершину v.

Нехай $G = (V, E)$ — орграф. Якщо $(u, v) \in E$, то вершина u називається *безпосереднім попередником* (*предком*) вершини v, а вершина v — *безпосередньо підлеглою вершині* u або *безпосередньо досяжною* з вершини u.

Число дуг $on(u)$, які виходять з вершини u, називається *степенем виходу вершини* u, а число дуг $in(u)$, які ведуть у вершину u, — *степенем входу вершини* u. Очевидно, що степінь вершини $n(u) = on(u) + in(u)$. Якщо $(\forall u \in V)\ on(u) \le 2$, то граф G називається **бінарним.** Цікаво, який вигляд має лема про рукостискання для орграфів. Оскільки кожна дуга орграфа із E «бере» участь у кожній із сум вигляду

$$\sum_{v \in V} in(v) \text{ і } \sum_{v \in V} on(v)$$

лише по одному разу, то маємо таке співвідношення:

$$\sum_{v \in V} in(v) = \sum_{v \in V} on(v) = |E|.$$

Отже, лема про рукостискання для орграфів набуває такого вигляду.

***Лема* про рукостискання для орграфів**. *Сума степенів входу всіх вершин орграфа дорівнює сумі степенів виходу всіх його вершин.*

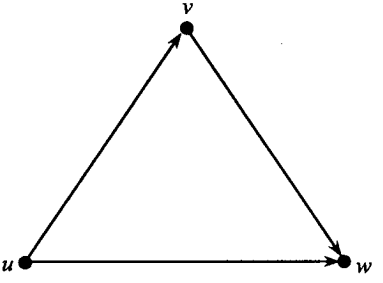
Цей результат часто називають *орлемою про рукостискання.*

Маршрутом, або **шляхом** l із вершини u у вершину v ($l = l(u, v)$) в орграфі G називається послідовність вершин $u = u_1, u_2, \ldots, u_k = v$ таких, що $(u_i, u_{i+1}) \in E$, $i = 1, 2, \ldots, k - 1$. При цьому вершину u називають *початковою*, а v — *кінцевою вершиною* шляху l і говорять, що u досяжна з v по шляху $l(u, v)$ або що шлях l веде з вершини u у вершину v. Дуги, якими з'єднані вершини маршруту, називаються ребрами маршруту. Маршрут називається *орланцюгом*, якщо всі дуги цього маршруту різні між собою, і *простим орланцюгом*, якщо всі вершини маршруту, можливо крім першої і останньої вершин, різні між собою. Орланцюг називається *замкнутим*, якщо перша і остан-

ня його вершини збігаються між собою. **Цикл** — це простий замкнутий орланцюг.

Якщо u — вершина графа G, для якої $on(u) = 0$ ($in(u) = 0$), то така вершина, як і раніше, називається *кінцевою*, або *висячою*, *вершиною*, або *стоком* (*джерелом*) графа G.

Приклад 5.11.14



Вершина u є джерелом, вершина w — стоком. ♠

Нехай R — відношення безпосередньої досяжності на множині вершин V, а R_t — його транзитивне замикання. Очевидно, що відношення досяжності збігається з відношенням R_t.

Визначення 133. *Орграф G називається* **сильно зв'язним**, *якщо $(\forall u, v \in V) u R_t v$. Зв'язний орграф називається* **ейлеровим орграфом**, *якщо в ньому існує замкнутий орланцюг, який проходить через кожну дугу графа.*

Теорема 148. *Зв'язний орграф є ейлеровим тоді і тільки тоді, коли $in(v) = on(v)$ для довільної його вершини.*

Доведення аналогічне доведенню аналогічної теореми для неорієнтованих графів.

Визначення 134. *Орграф G називається* **гамільтоновим**, *якщо в ньому існує цикл, що проходить через кожну з його вершин. Орграф називається* **напівгамільтоновим**, *якщо в ньому існує простий орланцюг, який проходить через кожну з його вершин.*

Логічно поставити запитання, а чи узагальнюється теорема Дірака на гамільтонові орграфи? Одне таке узагальнення належить Гуйя-Урі.

Теорема 149 **(Гуйя-Урі).** *Нехай G — сильно зв'язний орграф порядку n. Якщо $on(v) \geq n/2$ і $in(v) \geq n/2$ для довільної його вершини v, то G є гамільтоновим графом.*

Доведення даної теореми досить складне і виходить за рамки цього посібника.

Визначення 135. *Ациклічним орграфом називається орграф, у якого немає циклів.*

Орієнтованим деревом називається ациклічний орграф, який задовольняє такі умови:

1) $(\exists v_0 \in V)\ in(v_0) = 0$. *Вершина* v_0 *називається* **коренем**;
2) $(\forall v \neq v_0)\ in(v) = 1$, де $v \in V$;
3) $(\forall v \neq v_0)$ *існує шлях* $v_0, v_1, \ldots, v_n = v$, *причому єдиний.*

Нижче, якщо не обумовлено супротивне, під деревом буде розумітися орієнтоване дерево.

Орієнтований ліс — це орграф, який складається з кількох дерев. Якщо (u, v) — ребро дерева T, то вершина u називається **батьком**, а v — **сином** вершини u. Якщо $on(u) = 0$, то u називається **листком** у дереві. Вершина u разом з усіма своїми нащадками складає піддерево дерева T. При цьому u є коренем цього піддерева.

Теорема 150. *Орграф* $G = (V, E)$ *є орієнтованим деревом, якщо*

а) граф G *— зв'язний і існує* $v_0 \in V$, *яка не має предків, і для довільної вершини* $u \in V$ $in(u) = 1$;

б) граф G *має вершину* v_0, *з якої досяжна довільна інша вершина* u, *причому шлях, який з'єднує* v_0 *і* u, *єдиний*;

в) граф G *має вершину* v_0, *яка не має предків, а всі інші вершини мають тільки одного безпосереднього предка і з* v_0 *досяжна довільна інша вершина.*

Доведення пропонуються читачеві як прості вправи.

Упорядкованим деревом будемо називати дерево, у якого множина синів кожної вершини упорядкована.

Бінарне дерево T називається **повним**, якщо для кожної вершини u із V глибини, меншої від k, $on(u) = 2$, і якщо глибина u дорівнює k, то $on(u) = 0$, де під **глибиною вершини** u розуміють довжину шляху з кореня в цю вершину.

Висота вершини в дереві T — це довжина найдовшого шляху із заданої вершини у вершину-листок цього дерева, досяжний з даної вершини. **Висотою дерева** T називається висота його початкової вершини.

Рівень вершини — це різниця висоти дерева і глибини цієї вершини.

5.11.2. Бінарні відношення і орграфи

З визначення орграфа випливає, що множину ребер E можна розглядати як бінарне відношення на множині V. Отже, кожне бінарне відношення R на множині V може розглядатись як орграф $G(V, R)$, якщо за множину ребер графа взяти множину R. Очевидно, що справедливе і обернене твердження: кожний орграф $G = (V, E)$ задає деяке бінарне відношення R на множині V. Дійсно, якщо визначити бінарне відношення так, що два елементи u і v із множини V знаходяться у відношенні R між собою тоді і тільки тоді, коли $(u, v) \in E$, то відношення R і буде шуканим.

Таким чином, між бінарними відношеннями, визначеними на деякій скінченній множині V, і орграфами, для яких множина V служить множиною вершин, має місце взаємно однозначна відповідність.

Розглянемо графи, які відповідають відношенню еквівалентності.

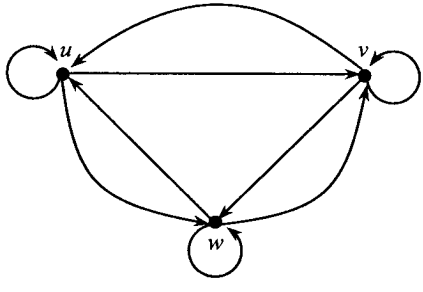
Нехай R — відношення еквівалентності на V. Із властивостей відношення R випливає, що V розбивається на класи еквівалентності V_1, V_2,..., V_k і в силу рефлексивності, симетричності та транзитивності відношення R маємо

1) vRv, тобто $(v, v) \in E$;

2) якщо uRv, то vRu, і тому $(u, v) \in E$ і $(v, u) \in E$;

3) якщо (uRv), (vRw), то (uRw), і тому $(u, w) \in E$.

Іншими словами, довільний елемент у класі еквівалентності має бути зв'язаний з кожним елементом цього класу ребром, і навпаки. Наприклад, якщо деякий клас має три вершини u, v, w, то граф цього класу еквівалентності виглядатиме так:



Отже, відношення еквівалентності R на множині V породжує k графів $G_1 = (V_1, E_1)$,..., $G_k = (V_k, E_k)$ (для кожного класу свій граф), кожний із яких — компонент зв'язності.

5.11.3. Позначені графи і представлення термів

Нехай MV і ME — деякі множини, які будемо називати множинами позначок.

Визначення 136. *Граф $G = (V, E)$ називається* **позначеним** *за допомогою функцій f і g, якщо*

$$f: V \to MV;$$
$$g: E \to ME.$$

При цьому функція f називається **функцією позначок вершин**, *а функція g —* **функцією позначок ребер**.
Нехай $G = (V, E)$ і $H = (V', E')$ — позначені графи за допомогою функцій f, g і f', g' відповідно.

Визначення 137. *Графи G і H називаються* **еквівалентними**, *якщо вони ізоморфні, і коли h — цей ізоморфізм, то*

$$f(u) = f'(h(u)),$$
$$g((u, v)) = g'((h(u), h(v)),$$

тобто відповідні вершини і ребра мають одні й ті самі позначки.

Якщо функція $f(g)$ тотожна, тобто $MV = V$ і $f(u) = u$ ($= E$ і $g((u, v)) = (u,v)$), то граф має позначені тільки ребра (тільки вершини). У цьому випадку в практичних застосуваннях функція $f(g)$ просто не береться до уваги.

Розглянемо тепер конструкцію позначеного графа, яка зручна для задання термів.

Нехай $G = (V, E)$ — орграф, $G' = (A, \Omega)$ — деяка алгебра і N — множина натуральних чисел. Покладемо $MV = \Omega$.

Складеним об'єктом називається четвірка $S = (V, L, f, v_0)$, де V — множина вершин, $v_0 \in V$ — початкова вершина, $f: V \to A \cup Q$ — функція позначок, $L \subseteq V \times V \times N$ — множина упорядкованих зв'язків, причому, якщо $f(v) = \omega^n \in \Omega$, то $|L \cap (v, V, N)| \le n$,

$$|L \cap (v, V, i)| \le \begin{cases} 1, & \text{якщо} \le n; \\ 0, & \text{якщо} > n. \end{cases}$$

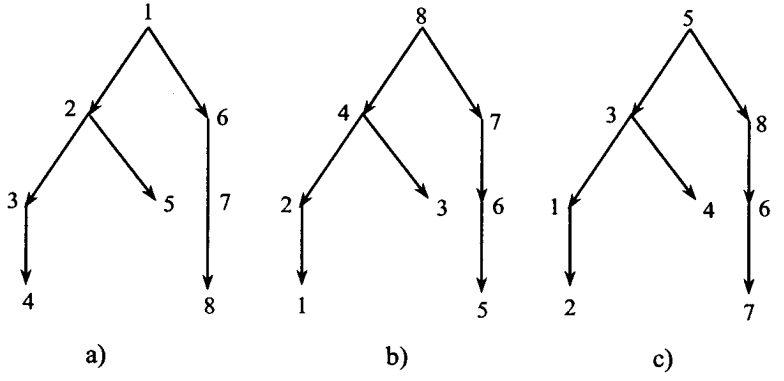
Нехай R_t — транзитивне замикання відношення безпосередньої досяжності. Складений об'єкт називається **ініціальним**, якщо для довільної вершини $v \ne v_0$ істинно $v_0 R_t v$. Нижче будуть розглядатися лише ініціальні складені об'єкти.

Умовимося надалі називати складений об'єкт просто складеним.

Граф $G = (V, E)$ буде називатися *графом складеного S*. Складений називається деревом, якщо його граф є деревом. Якщо $f(v) = a \in A$ або $f(v) = \omega$, де ω — нульарна операція із Ω, то $f(v)$ називають **первинним складеним об'єктом**. Кожна вершина складеного S породжує складений об'єкт, початковою вершиною якого є вона сама. Складений $S_{v'}$, який породжується вершиною v', називається i-м аргументом складеного S_v ($Arg(v, i) = v'$), що породжений вершиною v, якщо існує $i \in N$ таке, що $(v, v', i) \in L$. Скориставшись відношенням безпосередньої досяжності, операцію Arg можна узагальнити на всю множину V, якщо покласти для $v_1, v_2, \ldots, v_n \in V$ таких, що $Arg(v_k, i_{k+1}) = v_{k+1}$, $Arg(v, i_1, \ldots, i_k) = v_{k+1}$ $(k = 1, 2, \ldots, n-1)$ Позначкою складеного називається позначка його початкової вершини.

Велика кількість задач як для графів, так і для дерев, вимагають обходу вершин графа (дерева) в деякому порядку. Найбільш поширені три способи обходу вершин графів:

а) прямий; б) обернений; в) внутрішній.



a) b) c)

Частіше за інші використовується прямий порядок обходу вершин складеного, який ще називають обходом **зліва направо і зверху вниз.** Суть цього способу обходу полягає у виконанні таких дій:

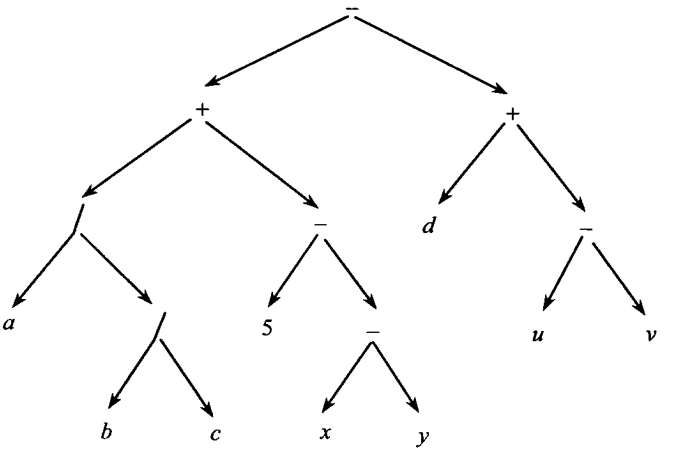
1) відвідати початкову вершину складеного — v_0;

2) відвідати найлівіший аргумент v' вершини v, який ще не відвідувався ($(v, v', i) \in L$).

Відповідно до цього порядку нумеруються і вершини складеного. При цьому вважається, що номер вершини v ототожнюється із самою вершиною. Завдяки такому ототожненню складений об'єкт можна задавати за допомогою масиву (масиву вершин) у пам'яті обчислювальної машини.

Складені об'єкти являють собою зручний спосіб для задання термів.

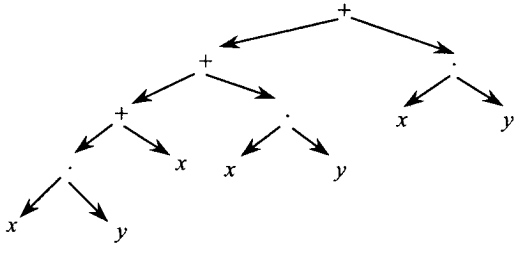
Приклад 5.11.15. 1. Терм $t = (((a/(b/c)) + (5 - (x - y)) - (d + (u - v)))$ задається таким складеним об'єктом:



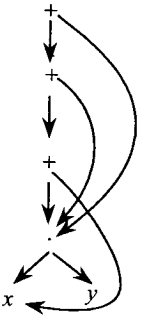
Складений, що представляє терм t

Рис. 5.11.27

2. Терм $t = (((x \cdot y) + x) + x \cdot y) + x \cdot y)$ задається таким складеним об'єктом:



Складений терма t



Ациклічний складений терма t

Зауважимо, що другий спосіб представлення терма набагато економніший, оскільки граф його складеного має лише 6 вершин і 8 дуг, у той час як перше представлення має на 7 вершин і 4 дуги більше. ♠

На завершення даного розділу зазначимо, що багато практичних алгоритмів на графах і деревах описано в монографіях [13, 11].

5.12. Контрольні питання, задачі і вправи

Контрольні питання

1. Що таке а) граф, б) мультиграф, в) псевдограф?
2. Що таке а) платонові графи, б) регулярний граф, в) повний граф, г) пустий граф, д) зв'язний граф, є) орієнтований граф?
3. Що таке а) двочастковий граф, б) повний двочастковий граф, в) k-частковий граф?
4. Яка різниця між а) графом і орграфом, б) графом і мультиграфом, в) графом і псевдографом?
5. Чи буде двочастковим графом а) дерево, б) ациклічний граф (орграф)?
6. Що означає запис $X_p(G) = 5$?
7. Чи кожний граф можна розфарбувати 5-ма фарбами?
8. Скільки двочасткових графів можна побудувати на множині, яка складається з чотирьох елементів?
9. Чи існує регулярний граф степеня 3 з трьома вершинами?
10. Побудуйте орграф, що відповідає відношенню еквівалентності, яке задане на трьохелементній множині з єдиним класом еквівалентності.

Задачі і вправи

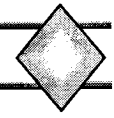
1. Довести наслідок 40.
2. Упевніться в тому, що двом різним деревам T і T' з n вершинами відповідають різні списки L і L', які будувалися в теоремі Келі.
3. Побудуйте транзитивне замикання відношення

$$R = \{(a, b), (a, c), (b, d), (d, a), (b, c)\},$$

яке задано на множині $A = \{a, b, c, d\}$.

4. Знайдіть число маршрутів довжини 3, що з'єднують вершини a і c для відношення R з прикладу 3.
5. Нехай дано дерево $T = (\{1, 2, 3, 4\}, \{(1, 2), (1, 3), (3, 4)\})$. Побудуйте його матрицю суміжностей і матрицю досяжностей. Які властивості має дана матриця? Чому дорівнює четвертий степінь даної матриці?
6. Побудуйте граф для відношення часткового порядку на деякій множині V.

6 МАТЕМАТИЧНА ЛОГІКА



Основним об'єктом вивчення в даному розділі є формальні логічні мови, які складають один із розділів сучасної математики, що називається **математична логіка.** Виникнення математичної логіки як математичної дисципліни пов'язують передусім зі строгим обґрунтуванням основ математики і, зокрема, теорії множин як фундаменту цих основ. Але останнім часом математична логіка все активніше використовується як мова специфікацій, логічна мова для верифікації властивостей програмних систем та їх математичних моделей.

Нижче розглядаються найуживаніші в застосуваннях формальні логічні мови, такі як логіка висловлювань, логіка предикатів першого порядку і лінійна темпоральна логіка та методи доведення теорем і перевірки виконуваності формул для цих логік.

6.1. Числення висловлювань

Числення висловлювань, яке розглядається нижче, буде для нас тим полігоном, на якому демонструватимуться основні прийоми пошуку доведень теорем та виконуваності формул у формальних логічних теоріях. Крім того, це числення входить як складова до більшості формальних логічних мов і, отже, привносить свої основні властивості у ці формальні мови.

Неформально висловлюванням називається довільне просте розповідне речення, яке щось стверджує про наше оточення і яке можна інтерпретувати як істинне або хибне, і яке не може бути одночасно і істинним, і хибним. Таке висловлювання називається простим, тобто висловлювання є простим, якщо воно не є збудованим з інших висловлювань. Висловлювання, збудоване згідно з певними правилами із простих, називається складним. Складне висловлювання, як і просте, також може бути істинним або хибним, але не тим і тим одночасно.

Розглянемо точні формальні означення.

6.1.1. Синтаксис і семантика числення висловлювань

Синтаксис. В численні висловлювань для позначення простих висловлювань будемо вживати алфавіт: $Al = \{A, B, C,..., A_1,...\}$ разом з такими спеціальними символами, які називаються логічними зв'язками і застосовуються до побудови складних висловлювань: $\{\neg$ («ні», заперечення), $\vee$ («або», альтернатива не виключається — диз'юнкція), $\wedge$ («і», кон'юнкція), $\rightarrow$ («імплікує», умовне висловлювання), $\leftrightarrow$ («тоді і тільки тоді, коли», еквівалентність)$\}$.

Прості висловлювання A, B,..., $\in Al$ називаються **атомарними формулами.**

Приклад 6.1.1 Наведені нижче речення є висловлюваннями:

a) $2 + 2 = 4$,
b) $2 + 3 = 7$,
c) Якщо множина A має n елементів, то $B(A)$ має 2^n елементів,
d) Юлій Цезар був президентом США.

Наступні речення не є висловлюваннями:

a) Це твоє, чи моє місце?
b) Для чого важливою є математична індукція?
c) $x - y = y - x$. ♠

Використовуючи індукцію, наведемо формальне означення синтаксису числення висловлювань (тобто правил, за якими будуються складні висловлювання з атомарних (простих)), що дає можливість вирізняти серед усіх слів у алфавіті Al слова, які будемо називати **правильно побудованими формулами.**

***Визначення 138* (Визначення синтаксису).** *Правильно побудованими формулами (ППФ) є:*

1) формули атомарні;
2) якщо A є ППФ, то $\neg A$ теж є ППФ;
3) якщо A і B є ППФ, то $A \vee B$, $A \wedge B$, $A \rightarrow B$, $A \leftrightarrow B$ теж є ППФ;
4) ППФ є тільки ті формули, які побудовані за правилами 1—3.

Семантика. Якщо A є атомарною формулою, то нехай $h(A)$ означає логічне значення формули A. Функцію h будемо називати **інтерпретацією** або **функцією логічних значень.** Отже,

$$h : Al \rightarrow \{0, 1\}.$$

Якщо A є складним висловлюванням, то коли $A = \neg A'$, то $h(A) = h(\neg A') = \neg h(A')$, а якщо $A = B \bigcirc C$, то $h(A) = h(B \bigcirc C) = h(B) \bigcirc h(C)$, де $\bigcirc \in \{\wedge, \vee, \rightarrow, \leftrightarrow\}$.

Звідси випливає, що логічне значення складного висловлювання цілком залежить від логічних значень атомарних формул, з яких побудоване це висловлювання за допомогою логічних зв'язок.

***Визначення 139* (Визначення семантики).** *Якщо A і B є ППФ, то*
a) $h(\neg A) = 1$ *тоді і тільки тоді, коли* $h(A) = 0$ *і* $h(\neg A) = 0$ *тоді і тільки тоді, коли* $h(A) = 1$;
b) $h(A \vee B) = 1$ *тоді і тільки тоді, коли* $h(A) = 1$ *або* $h(B) = 1$;
c) $h(A \wedge B) = 1$ *тоді і тільки тоді, коли* $h(A) = 1$ *і* $h(B) = 1$;
d) $h(A \to B) = 1$ *тоді і тільки тоді, коли* $h(A) \le h(B)$;
e) $h(A \leftrightarrow B) = 1$ *тоді і тільки тоді, коли* $h(A) = h(B)$.

З цього означення випливає, що коли A і B є ППФ, то для нижченаведених формул маємо такі таблиці логічних значень, які називають **таблицями істинності**:

A	$\neg A$	A	B	$A \vee B$	$A \wedge B$	$A \to B$	$A \leftrightarrow B$
1	0	1	1	1	1	1	1
0	1	1	0	1	0	0	0
		0	1	1	0	1	0
		0	0	0	0	1	1

За допомогою таких таблиць можна побудувати таблицю істинності (яку часто називають матрицею істинності) довільної складної формули числення висловлювань, яка побудована з атомарних формул A, B, C,... . Така таблиця включає логічні значення складових підформул, які залежать від логічних значень атомарних формул A, B, C... . Атомарні формули A, B, C,... називаються **пропозиційними змінними** складної формули. Логічне значення складного висловлювання, таким чином, можна обчислити шляхом обчислення значень підформул, з яких воно збудоване.

Приклад 6.1.2. Формула $P = (A \wedge B) \vee \neg (A \to B)$ збудована з двох простих формул A і B. Отже, існує тільки чотири можливі комбінації логічних значень для A і B (див. табл.).

ТАБЛИЦЯ ІСТИННОСТІ ДЛЯ ФОРМУЛИ *P*

1	2	3	4	5	6
A	*B*	$A \wedge B$	$A \to B$	$\neg(A \to B)$	$(A \wedge B) \vee \neg(A \to B)$
1	1	1	1	0	1
1	0	0	0	1	1
0	1	0	1	0	0
0	0	0	1	0	0

Значення в рядках 3 і 4 знайдені на основі значень у стовпчиках 1 і 2. Значення у стовпчику 5 визначається на основі значень у стовпчику 4. Значення у стовпчику 6 (логічне значення всієї формули *P*) обчислене за відповідними значеннями у стовпчиках 3 і 5. ♠

Визначення 140. *Будемо вважати, що формула A істинна при деякій інтерпретації h тоді і тільки тоді, коли* $h(A) = 1$. *Якщо* $h(A) = 0$, *то говоримо, що формула A хибна при інтерпретації h.*

Визначення 141. *Формула A називається* **тавтологією (суперечністю),** *якщо вона істинна (хибна) при будь-якій інтерпретації h.*

Визначення 142. *Якщо формула* $A \to B$ *є тавтологією, то говорять, що формула B випливає з формули A, або що висловлювання B є логічним наслідком висловлювання A. Якщо висловлювання* $A \leftrightarrow B$ *є тавтологією, то говорять, що формули A і B логічно еквівалентні.*

Користуючись таблицями істинності, завжди можна з'ясувати, є чи ні дана формула тавтологією.

Приклад 6.1.3. 1) Чи є формула $((A \to B) \to B) \to B$ тавтологією?

Розв'язок. Згідно з таблицею істинності для зв'язки $\to$ отримуємо

A	*B*	$A \to B$	$(A \to B) \to B$	$((A \to B) \to B) \to B$
1	0	0	1	0
1	1	1	1	1
0	1	1	1	1
0	0	1	0	1

і, як видно з останнього стовпчика, дана формула не є тавтологією, оскільки при інтерпретації $h(A) = 1$ і $h(B) = 0$ вона хибна.

2) Формули A і $A \to B$ є тавтологіями, довести, що формула B також є тавтологією.

Доведення.

A	B	$A \to B$
1	1	1
1	0	1

Якщо формула B хибна, то і формула $A \to B$ також має бути хибною, а це суперечить тому, що остання формула є тавтологією. Отже, B також мусить бути тавтологією.

3) Чи є формула $A \to A$ тавтологією?

Розв'язок. Відповідно до таблиці істинності для зв'язки $\to$ отримуємо

A	$A \to A$
1	1
0	1

Отже, дана формула є тавтологією. ♠

За допомогою таблиць істинності легко переконатися, що для логічних зв'язок $\neg$, $\vee$, $\wedge$ мають місце закони комутативності, асоціативності, дистрибутивності, поглинання, де Моргана і т. д. Дійсно, для зв'язки $\wedge$ маємо

A	B	$A \wedge B$	$B \wedge A$
0	0	0	0
0	1	0	0
1	0	0	0
1	1	1	1

Звідси випливає, що $A \wedge B \leftrightarrow B \wedge A$, оскільки формули $A \wedge B$ і $B \wedge A$ мають однакові логічні значення при кожній інтерпретації h.

Покажемо, що $A \leftrightarrow A \vee 0$:

A	$A \vee A$
0	0
1	1

Аналогічно до попереднього, з вищенаведеної таблиці випливає висновок, що $A \leftrightarrow A \vee 0$.

Так само можна переконатися у справедливості законів де Моргана:

A	B	$\neg A \wedge \neg B$	$\neg(A \vee B)$
0	0	1	1
0	1	0	0
1	0	0	0
1	1	0	0

A	B	$\neg A \vee \neg B$	$\neg(A \wedge B)$
0	0	1	1
0	1	1	1
1	0	1	1
1	1	0	0

або $\neg(A \vee B) \leftrightarrow \neg A \wedge \neg B$ і $\neg(A \wedge B) \leftrightarrow \neg A \vee \neg B$.

Доведення решти законів пропонуються як вправи.

6.1.2. Повні системи зв'язок

Довільна формула з n пропозиційними змінними є n-арною функцією логічних значень. Еквівалентні формули мають однакові логічні значення. Звідси виникає питання: чи всі функції логічних значень можна отримати в такий спосіб?

Теорема 151. *Довільна функція логічних значень є деякою формулою числення висловлювань, яку можна побудувати із атомарних формул лише з використанням логічних зв'язок* $\neg$, $\wedge$, $\vee$.

Доведення. Нехай $f(x_1, x_2, \ldots, x_n)$ є даною функцією логічних значень. Очевидно, що f можна представити за допомогою таблиці істинності з 2^n рядками, де кожний рядок представляє певну множину логічних значень змінних (інтерпретацію) $x_1, x_2, \ldots, x_n$, а також відповідне значення $f(x_1, x_2, \ldots, x_n)$. Перенумеруємо рядки цієї таблиці за допомогою натуральних чисел 1, 2,..., 2^n. Нехай для кожного $i = 1, 2, \ldots, 2^n$, C_i означає формулу $U_1^i \wedge U_2^i \wedge \ldots \wedge U_n^i$, де U_j^i означає A_j, якщо в i-тому рядку таблиці істинності x_j дорівнює 1, або $\neg A_j$, якщо x_j дорівнює 0. Нехай D є диз'юнкція всіх формул C_i таких, що f у i-тому рядку дорівнює 1.

Якщо таких рядків немає, то f завжди хибна і для такої функції можна використати формулу вигляду $A_1 \wedge \neg A_1$. Покажемо, що таблиця істинності для D еквівалентна f. Нехай в k-тому рядку задана деяка множина логічних значень для атомарних формул $A_1, A_2, \ldots, A_n$ разом зі значенням функції f, що відповідає цій множині. Тоді тільки формула C_k є істинною на цій множині логічних значень, у той час як решта формул C_i є хибними на цій множині.

Якщо C_k істинна, то C_k є елементом D, а тоді D теж істинна. Якщо ж f хибна в даному рядку, то C_k не є елементом D, і оскільки решта формул C_i в даному рядку хибні, то і D теж хибна в цьому рядку. ■

Приклад 6.1.4. 1) Нехай дано функцію логічних значень $f(x_1, x_2)$:

x_1	x_1	$f(x_1, x_2)$
1	1	0
0	1	1
1	0	1
0	0	1.

Згідно з доведенням теореми 151, будуємо формулу C_2, C_3 і C_4, оскільки для цих інтерпретацій функція $f(x_1, x_2)$ істинна:

$$C_2 = \neg A_1 \wedge A_2,\ C_3 = A_1 \wedge \neg A_2 \text{ і } C_4 = \neg A_1 \wedge \neg A_2.$$

Нарешті будуємо диз'юнкцію, яка має вигляд

$$D = C_2 \vee C_3 \vee C_4 = (\neg A_1 \wedge A_2) \vee (A_1 \wedge \neg A_2) \vee (\neg A_1 \wedge \neg A_2).$$

Застосовуючи тотожні перетворення, отриману формулу можна спростити до такого вигляду:

$$\begin{gathered} D = (\neg A_1 \wedge A_2) \vee (A_1 \wedge \neg A_2) \vee (\neg A_1 \wedge \neg A_2) = \\ = (\neg A_1 \wedge A_2) \vee ((A_1 \vee \neg A_1) \wedge \neg A_2) = (\neg A_1 \wedge A_2) \vee \neg A_2 = \\ = (\neg A_1 \vee \neg A_2) \wedge (A_2 \vee \neg A_2) = (\neg A_1 \vee \neg A_2) \wedge 1 = \neg A_1 \vee \neg A_2. \end{gathered}$$

2) Нехай дано функцію логічних значень $f(x_1, x_2)$:

x_1	x_2	$f(x_1, x_2)$
1	1	1
0	1	1
1	0	0
0	0	1.

Аналогічно до попереднього прикладу, отримуємо

$$C_1 = A_1 \wedge A_2,\ C_2 = \neg A_1 \wedge A_2,\ C_4 = \neg A_1 \wedge \neg A_2.$$

У результаті маємо:

$$D = C_1 \vee C_2 \vee C_4 = (A_1 \wedge A_2) \vee (\neg A_1 \wedge A_2) \vee (\neg A_1 \wedge \neg A_2).$$

Використовуючи закони асоціативності, комутативності і дистрибутивності, отримуємо:

$$\begin{gathered} D = (\neg A_1 \wedge \neg A_2) \vee ((A_1 \vee \neg A_1) \wedge A_2) = \\ = (\neg A_1 \wedge \neg A_2) \vee A_2 = (\neg A_1 \vee A_2) \wedge (\neg A_2 \vee A_2) = \neg A_1 \vee A_2. \end{gathered}$$

Зауважимо, що $f(x_1, x_2)$ є функцією логічних значень для формули $A_1 \to A_2$, тобто $A_1 \to A_2 \leftrightarrow \neg A_1 \vee A_2$.

3) Нехай дано функцію логічних значень $f(x_1, x_2)$:

x_1	x_2	$f(x_1, x_2)$
1	1	1
0	1	0
1	0	0
0	0	1.

Тоді отримуємо $C_1 = A_1 \wedge A_2$, $C_2 = \neg A_1 \wedge \neg A_2$. Отже, маємо

$$D = (\neg A_1 \wedge \neg A_2) \vee (A_1 \wedge A_2) = (\neg A_1 \vee (A_1 \wedge A_2)) \wedge (\neg A_2 \vee (A_1 \wedge A_2)) =$$
$$= ((\neg A_1 \vee A_1) \wedge (\neg A_1 \vee A_2)) \wedge ((\neg A_2 \vee A_1) \wedge (\neg A_2 \vee A_2)) =$$
$$= (\neg A_1 \vee A_2) \wedge (\neg A_2 \vee A_1).$$

Зауважимо, що $f(x_1, x_2)$ є функцією логічних значень для формули $A_1 \leftrightarrow A_2$, тобто

$$(A_1 \leftrightarrow A_2) \leftrightarrow (\neg A_1 \vee A_2) \wedge (\neg A_2 \vee A_1) \leftrightarrow (A_1 \to A_2) \wedge (A_2 \to A_1). \spadesuit$$

Теорема 152. *Довільна функція логічних значень f є деяким висловлюванням, яке можна записати, використовуючи тільки одну з таких пар логічних зв'язок*: $(\neg, \wedge)$, $(\neg, \vee)$, $(\to, \neg)$.

Доведення випливає з теореми 151, прикладу 6.1.4 (приклади 2) і 3)) і законів де Моргана. ■

6.1.3. Аксіоматична система для числення висловлювань

Формалізація тієї, чи іншої логічної мови необхідна для того, щоб можна було формалізувати і, по можливості, автоматизувати процес побудови правильних логічних наслідків у цій формальній мові. З цією метою і вводиться аксіоматизація логічних мов і, зокрема, числення висловлювань. Перейдемо до розгляду цієї аксіоматизації.

Формальна логічна теорія *Th* вважається визначеною, якщо задані:

1. Деякий **скінченний** або **зліченний алфавіт**, елементи якого є символами теорії *Th*; скінченні послідовності символів цього алфавіту називають **виразами теорії *Th***.

2. Підмножина виразів теорії *Th*, яка називається **множиною формул теорії** *Th*; часто існує процедура, за допомогою якої завжди можна з'ясувати, є чи ні даний вираз формулою.

3. Підмножина формул, елементи якої називаються **аксіомами теорії** *Th*; якщо існує процедура, за допомогою якої можна з'ясувати, є чи ні дана формула аксіомою, то теорія *Th* називається **аксіоматичною теорією**.

4. Скінченна множина R_1, R_2,..., R_n відношень на множині формул, елементи якої називаються **правилами виведення**; для довільного відношення R_i існує $j \in N^+$ таке, що для даної множини формул A_1, A_2,..., A_i і довільної формули A можна з'ясувати, є чи ні відношення R_i істинним для даних формул A_1, A_2,..., A_i і формули A, і якщо $(A_1, A_2,..., A_i, A) \in R_i$, то A називається **безпосереднім наслідком** формул A_1, A_2,..., A_i за правилом $R_i((A_1,..., A_i)(R_i) \vdash A)$.

Визначення 143. **Доведенням у теорії Th** *називається довільна скінченна послідовність формул A_1, A_2,..., A_n цієї теорії така, що кожна формула A_i є або аксіомою або безпосереднім наслідком попередніх формул цієї послідовності за одним із правил виведення.*

Формула A називається **теоремою теорії** *Th*, якщо існує доведення в *Th* для A таке, що останнім елементом у цьому доведенні є формула A. Таке доведення називається доведенням формули A в теорії *Th*.

Якщо існує алгоритм, за допомогою якого можна з'ясувати, є чи ні дана формула A теоремою теорії *Th*, то теорія *Th* називається **розв'язуваною**, у протилежному випадку **нерозв'язуваною.**

Формула A називається логічним наслідком множини формул Г в теорії *Th* тоді і тільки тоді, коли існує така послідовність формул A_1, ..., A_n, що $A_n = A$ і кожна з формул A_i є або аксіомою, або елементом множини Г, або безпосереднім наслідком деяких попередніх формул за одним із правил виведення. Позначається це як $\Gamma \vdash A$.

Визначення 144. *Послідовність формул A_1,..., A_n називається* **доведенням формули** *A із множини формул* Г, *а елементи множини* Г — **гіпотезами** *або* **припущеннями.**

Визначення аксіоматичної системи. Формальна аксіоматична теорія *ЧВ* для числення висловлювань визначається таким чином:

1. Алфавітом теорії *ЧВ* є алфавіт AL і множина символів $\neg$, $\rightarrow$, (,). Символи $\neg$, $\rightarrow$ називаються логічними зв'язками, а літери алфавіту A1 **пропозиційними змінними** або **(пропозиційними) літерами**.

2. Усі літери алфавіту A1 є формулами теорії *ЧВ*. Якщо A і B є формулами *ЧВ*, то $(\neg A)$, $(A \rightarrow B)$ також є формулами теорії *ЧВ*.

3. Для довільних формул A, B, C теорії *ЧВ* формули:

A1) $A \rightarrow (B \rightarrow A)$,

A2) $((A \rightarrow (B \rightarrow C)) \rightarrow ((A \rightarrow B) \rightarrow (A \rightarrow C))$,

A3) $(\neg B \rightarrow \neg A) \rightarrow ((\neg B \rightarrow A) \rightarrow B)$

є схемами аксіом.

4. Єдиним правилом виведення є правило **modus ponens** (MP):

$$A, A \rightarrow B \vdash B$$

Решту логічних зв'язок можна ввести за допомогою вже відомих формул:

$A \vee B \leftrightarrow \neg A \to B$

$A \wedge B \leftrightarrow \neg(A \to \neg B)$ $((A \wedge B) \leftrightarrow \neg(\neg A \vee \neg B) \leftrightarrow \neg(A \to \neg B))$

$(A \leftrightarrow B) \leftrightarrow (A \to B) \wedge (B \to A)$.

Зауважимо, що нескінченна множина аксіом теорії ЧВ задається за допомогою лише трьох **схем аксіом** А1), А2), А3), кожна з яких породжує нескінченну множину аксіом. Скінченність числа схем аксіом дає можливість для довільної формули легко перевірити, є чи ні вона аксіомою, і, отже, теорія ЧВ є аксіоматичною теорією.

Приклад 6.1.5. 1) Довести, що формула $A \to A$ є теоремою ЧВ **(Рефлексивність імплікації (РІ))**

Доведення. Підставимо формулу $A \to A$ у схему аксіом А2): $(A \to \to ((A \to A) \to A)) \to ((A \to (A \to A)) \to (A \to A))$. Оскільки $(A \to \to ((A \to A) \to A))$ є аксіомою А1), то за правилом МР маємо: $(A \to (A \to A)) \to (A \to A)$. Формула $A \to (A \to A)$ є аксіомою А1) і тоді за правилом МР маємо $A \to A$. Отже, формула $A \to A$ є теоремою *ЧВ*.

2) Довести, що $((\neg A \to A) \to A)$ є теоремою ЧВ.

Доведення. За аксіомою А3) маємо $(\neg A \to \neg A) \to ((\neg A \to A) \to A)$. За попередньою теоремою і правилом МР маємо $(\neg A \to A) \to A$.♠

Незалежність аксіом. Якщо X є деякою підмножиною множини аксіом Y, то ця підмножина називається незалежною, якщо для жодної з формул множини X не існує доведення за правилами виведення із множини аксіом $Y \setminus X$.

Теорема 153. *Кожна з аксіом А1)—А3) незалежна* [28, 29].

Доведення. **Незалежність** А1). Розглянемо такі таблиці:

A	$\neg A$	A	B	$A \to B$
0	1	0	0	0
1	1	1	0	2
2	0	2	0	0
		0	1	2
		1	1	2
		2	1	0
		0	2	2
		1	2	0
		2	2	0

За довільного розподілу значень 0, 1, 2 для атомарних формул, які входять у формулу A, ці таблиці дають можливість знайти відповідні значення формули A. Будемо називати формулу A **виділеною**, якщо вона завжди набуває значення 0 відповідно до цих таблиць.

Неважко переконатися в тому, що правило MP зберігає властивість виділеності, а також у тому, що довільна аксіома, отримана зі схем аксіом $A2$) чи $A3$), теж є виділеною формулою. Звідси випливає, що довільна формула, яка виводиться із аксіом $A2$) — $A3$) за допомогою правила MP, буде виділеною. Але формула

$$A \to (B \to A),$$

яка є окремим випадком аксіоми A1), не є виділеною, оскільки вона набуває значення 2, коли A набуває значення 1, а B приймає значення 2. Отже, аксіома A1) не залежить від аксіом A2) — A3).

Незалежність A2). Розглянемо такі таблиці:

A	$\neg A$	A	B	$A \to B$
0	1	0	0	0
1	1	1	0	0
2	0	2	0	0
		0	1	2
		1	1	2
		2	1	0
		0	2	1
		1	2	0
		2	2	0

Будемо називати формулу A **гротескною**, якщо вона завжди набуває значення 0 відповідно до цих таблиць. Неважко переконатися в тому, що правило MP зберігає властивість гротескності, а також у тому, що довільна аксіома, отримана зі схем аксіом $A1$) чи $A3$), теж є гротескною формулою. Звідси випливає, що довільна формула, яка виводиться із аксіом $A1$) і $A3$) за допомогою правила MP, буде гротескною. Але формула

$$(A \to (B \to C)) \to ((A \to B) \to (A \to C)),$$

яка є окремим випадком аксіоми $A2$), не є гротескною, оскільки вона набуває значення 2, коли A, B і C набувають відповідно значення 0, 0 і 1. Отже, аксіома $A2$) не залежить від аксіом $A1$) і $A3$).

Незалежність А3). Нехай A — довільна формула і $z(A)$ — формула, отримана з формули A шляхом стирання всіх входжень знака заперечення у формулі A. Як відомо, правило МР зберігає властивість формули A мати тавтологією $z(A)$. Очевидно також, що кожний окремий випадок A схеми аксіоми А1) чи А2) є такою формулою, що $z(A)$ тавтологія. Отже, довільна формула A, яка виводиться із аксіом А1)—А2) за допомогою правила МР, така, що $z(A)$ є тавтологією. Але формула

$$z(((\neg A \to \neg A) \to ((\neg A \to A) \to A))$$

набуває вигляду

$$(A \to A) \to ((A \to A) \to A)).$$

Отримана формула є окремим випадком аксіоми $A3$) і не є тавтологією, оскільки вона набуває значення 0, коли A набуває значення 0. Отже, аксіома $A3$) не залежить від аксіом $A1$) і $A2$). ■

Інші аксіоматизації числення висловлювань. Система аксіом $A1$)—$A3$) не єдина можлива аксіоматична система *ЧВ*. Існують і інші аксіоматичні системи числення висловлювань, дві з яких наводяться нижче.

Аксіоматизація Россера

Алфавітом теорії є алфавіт *Al*. Логічними зв'язками є $\wedge$, $\neg$, а також $A \to B$ як скорочення формули $\neg(A \wedge \neg B)$.

Схемами аксіом є такі формули:

RA1) $A \to (A \wedge A)$;

RA2) $(A \wedge B) \to A$;

RA3) $(A \to B) \to (\neg(B \wedge C) \to \neg(C \wedge A))$.

Єдиним правилом виведення є МР.

Аксіоматизація Гільберта-Аккермана

Алфавітом теорії є алфавіт *Al*. Логічними зв'язками є $\vee$, $\neg$, а також $A \to B$ як скорочення формули $\neg A \vee B$.

Схемами аксіом є такі формули:

HAA1) $(A \vee A) \to A$;

HAA2) $A \to (A \vee B)$;

HAA3) $(A \vee B) \to (B \vee A)$;

HAA4) $(B \to C) \to ((A \vee B) \to (A \vee C))$.

Єдиним правилом виведення є МР.

6.1.4. Теорема дедукції

Наведене нижче твердження дедукції є надзвичайно корисним твердженням числення висловлювань, оскільки воно складає додаткове правило виведення, яке часто суттєво скорочує доведення теорем у теорії ЧВ.

Теорема 154 (*Теорема дедукції*). *Якщо* Γ *є множиною формул, а* A *і* B *є формулами і* $\Gamma, A \vdash B$, *то* $\Gamma \vdash A \to B$. *Зокрема, якщо* $A \vdash B$, *то* $\vdash A \to B$.

Доведення. Нехай $B_1, B_2, \ldots, B_n$ є доведенням формули B з множини формул $\Gamma \cup \{A\}$, тобто $B_n = B$. Доведення ведеться методом математичної індукції за довжиною i ($1 \le i \le n$) доведення формули B. Якщо $i = 1$, то B_1 може бути:

a) елементом множини Γ, або

b) аксіомою, або

c) формулою A.

У випадках a) і b) справедливість теореми випливає з того, що з аксіоми *A1*) маємо $B_1 \to (A \to B_1)$. Потім за правилом MP одержуємо $\Gamma \vdash A \to B_1$.

У випадку c), коли $A = B_1$, доведення теореми випливає з того, що формула $A \to A$ є теоремою ЧВ. Отож $\vdash A \to A$.

Нехай $\vdash A \to B_k$ справедлива для всіх $k < i$, покажемо, що $\Gamma \vdash A \to$ $\to B_i$. Як і раніше, можливі такі випадки:

a) B_i є аксіомою,

b) B_i є елементом Γ,

c) $B_i = A$,

d) B_i є наслідком деяких формул B_j, B_m, де $j, m < i$, за правилом MP і B_m має вигляд $B_j \to B_i$.

Доведення випадків a)—c) нічим не відрізняється від доведення попередніх випадків a)—c) для $i = 1$. Використовуючи припущення індукції для випадку d), отримуємо

$$\Gamma \vdash A \to B_j \text{ і } \Gamma \vdash A \to (B_j \to B_i).$$

Згідно з аксіомою *A2*) маємо

$$(A \to (B_j \to B_i)) \to ((A \to B_j) \to (A \to B_i).$$

За правилом MP отримуємо

$$\Gamma \vdash (A \to B_j) \to (A \to B_i)$$

і знову за правилом MP $\Gamma \vdash A \to B_i$. ■

Застосуємо теорему дедукції для доведення основних теорем теорії ЧВ.

6.1.5. Основні властивості числення висловлювань

Основні властивості теорії ЧВ випливають з такого твердження.

Теорема 155. *Для довільних формул A, B, C числення висловлювань наступні формули є теоремами теорії ЧВ*:

a) $A \to B, B \to C \vdash A \to C$ **(Транзитивність імплікації (ТІ));**

b) $A \to (B \to C), B \vdash A \to C$;

c) $\neg\neg B \leftrightarrow B$ **(Закон подвійного заперечення (ЗПЗ));**

d) $\neg A \to (A \to B) \leftrightarrow \neg A \to (\neg A \vee B)$ **(Перше правило диз'юнкції (ПД1));**

e) $(\neg B \to \neg A) \leftrightarrow (A \to B)$ **(Правило контрапозиції);**

f) $A \to (\neg B \to \neg(A \to B))$;

g) $(A \to B) \to ((\neg A \to B) \to B)$;

h) $\neg A \to B, A \to C \vdash \neg B \to C$ **(Правило резолюцій);**

i) $A \to B, A \to C \vdash A \to (B \wedge C)$ **(Перше правило кон'юнкції (ПК1));**

k) $\vdash (A \wedge B) \to A$ **(Друге правило кон'юнкції (ПК2));**

l) $A \to B, C \to B \vdash (A \vee C) \to B$ **(Друге правило диз'юнкції (ПД2)).**

Доведення.

a) $A \to B, B \to C \vdash A \to C$

(1) $A \to B$ — гіпотеза,
(2) $B \to C$ — гіпотеза,
(3) A — гіпотеза,
(4) B — MP з (3) і (1),
(5) C — MP з (4) і (2).

Отже, $A \to B, B \to C, A \vdash C$ і за теоремою дедукції отримуємо: $A \to B, B \to C \vdash A \to C$.

Доведене твердження свідчить, що імплікація, як бінарне відношення на множині формул ЧВ, є транзитивним відношенням.

b) $A \to (B \to C), B \vdash A \to C$

(1) $A \to (B \to C)$ — гіпотеза,
(2) B — гіпотеза,
(3) A — гіпотеза,
(4) $B \to C$ — MP з (3) і (1),
(5) C — MP з (2) і (4).

Тепер $A \to (B \to C)$, B, $A \vdash C$ і за теоремою дедукції отримуємо: $A \to (B \to C), B \vdash A \to C$.

$$c) \neg\neg B \leftrightarrow B$$
$$c1) \neg\neg B \to B$$

(1) $(\neg B \to \neg\neg B) \to ((\neg B \to \neg B) \to B)$ — аксіома А3),
(2) $\neg B \to \neg B$ — доведена теорема,
(3) $(\neg B \to \neg\neg B) \to B$ — теорема 155 b),
(4) $\neg\neg B \to (\neg B \to \neg\neg B)$ — аксіома А1),
(5) $\neg\neg B \to B$ — ТІ з (3), (4) (теорема 155а)).

$$c2) B \to \neg\neg B$$

(1) $(\neg\neg\neg B \to \neg B) \to ((\neg\neg\neg B \to B) \to \neg\neg B)$ — аксіома А3),
(2) $\neg\neg\neg B \to \neg B$ — теорема155 $c1$),
(3) $(\neg\neg\neg B \to B) \to \neg\neg B$ — МР з (2) і (1),
(4) $B \to (\neg\neg\neg B \to B)$ — аксіома А1),
(5) $B \to \neg\neg B$ — ТІ з (4) і (3),

На основі теорем $c1$) і $c2$) отримуємо $B \leftrightarrow \neg\neg B$ — **закон подвійного заперечення (ЗПЗ).**

$$d) \neg A \to (A \to B)$$

(1) $\neg A$ — гіпотеза,
(2) A — гіпотеза,
(3) $A \to (\neg B \to A)$ — аксіома А1),
(4) $\neg B \to A$ — МР із (2) і (3),
(5) $\neg A \to (\neg B \to \neg A)$ — аксіома А1),
(6) $\neg B \to \neg A$ — МР з (1) і (5),
(7) $(\neg B \to \neg A) \to ((\neg B \to A) \to B)$ — аксіома А3),
(8) $(\neg B \to A) \to B$ — МР із (6) і (7),
(9) B — МР із (4) і (8).

Отже, $\neg A, A \vdash B$ і за теоремою дедукції отримуємо $\vdash \neg A \to (A \to B)$.

Варто зауважити, що коли застосувати тотожність до зв'язки $\vee$, то дана формула набуває такого вигляду: $\neg A \to (\neg A \vee B)$. Якщо в цій формулі замінити $\neg A$ на A, то отримаємо таку формулу:

$$A \to (A \vee B),$$

яку будемо називати **першим правилом диз'юнкції (ПД1).**

$$e) (A \to B) \leftrightarrow (\neg B \to \neg A)$$
$$e1) (\neg B \to \neg A) \to (A \to B)$$

(1) $\neg B \to \neg A$ — гіпотеза,
(2) A — гіпотеза,

(3) $(\neg B \to \neg A) \to ((\neg B \to A) \to B)$ — аксіома А3),
(4) $A \to (\neg B \to A)$ — аксіома А1),
(5) $(\neg B \to A) \to B$ — МР з (1) і (3),
(6) $A \to B$ — ТІ з (4), (5),
(7) B — МР з (2) і (6).

Отже, $\neg B \to \neg A, A \vdash B$ і за теоремою дедукції отримуємо $\vdash (\neg B \to \to \neg A) \to (A \to B)$.

$e2)\ (A \to B) \to (\neg B \to \neg A)$

(1) $A \to B$ — гіпотеза,
(2) $(A \to B) \to ((A \to \neg B) \to \neg A)$ — аксіома А3),
(3) $(A \to \neg B) \to \neg$A — МР з (1) і (2),
(4) $\neg B \to (A \to \neg B)$ — аксіома А1),
(5) $\neg B \to \neg A$ — ТІ з (4) і (3).

Отже, $A \to B \vdash \neg B \to \neg A$ і за теоремою дедукції отримуємо $\vdash (A \to \to B) \to (\neg B \to \neg A)$.

З теорем *e*l) і *e*2) отримуємо: $A \to B \leftrightarrow \neg B \to \neg A$. Отримана теорема є основою для так званого методу доведення теорем від супротивного і називається **правилом контрапозиції**.

$f)\ A \to (\neg B \to \neg(A \to B))$

Оскільки $A, A \to B \vdash B$, то за теоремою дедукції маємо $A \to ((A \to \to B) \to B)$. З правила контрапозиції випливає, що $\vdash ((A \to B) \to B) \to \to (\neg B \to \neg(A \to B))$. Нарешті, за правилом транзитивності імплікації (теорема 155 а)) отримуємо $\vdash A \to (\neg B \to \neg(A \to B))$.

$g)\ (A \to B) \to ((\neg A \to B) \to B)$

(1) $A \to B$ — гіпотеза,
(2) $\neg A \to B$ — гіпотеза,
(3) $(A \to B) \to (\neg B \to \neg A)$ — теорема 155 е),
(4) $\neg B \to \neg A$ — МР з (1) і (3),
(5) $(\neg A \to B) \to (\neg B \to \neg\neg A)$ — теорема 155 е),
(6) $\neg B \to \neg\neg A$ — МР з (2) і (5),
(7) $(\neg B \to \neg\neg A) \to ((\neg B \to \neg A) \to B)$ — аксіома А3),
(8) $(\neg B \to \neg A) \to B$ — МР із (6) і (7),
(9) B — МР з (4) і (8).

Отже, $A \to B, \neg A \to B \vdash B$ і за теоремою дедукції маємо $(A \to B) \to \to ((\neg A \to B) \to B)$.

$h) \neg A \to B, A \to C \vdash \neg B \to C$

(Правило резолюцій)

(1) $\neg A \to B$ — гіпотеза,

(2) $A \to C$ — гіпотеза,
(3) $\neg B$ — гіпотеза,
(4) $(\neg A \to B) \to (\neg B \to A)$ — теорема 155 e),
(5) $\neg B \to A$ — MP з (1) і (4),
(6) A — MP з (3) і (5),
(7) C — MP із (6) і (2).

За теоремою дедукції маємо $\neg A \to B, A \to C \vdash \neg B \to C$.

Правило резолюцій часто записують у так званій альтернативній формі, використовуючи тотожності для зв'язок $\to$ і $\vee$:

$$A \vee B, \neg A \vee C \vdash B \vee C.$$

i) $A \to B, A \to C \vdash A \to (B \wedge C)$

(Перше правило кон'юнкції (ПК1))

(1) $A \to B$ — гіпотеза,
(2) $A \to C$ — гіпотеза,
(3) A — гіпотеза,
(4) B — MP з (3) і (1),
(5) C — MP з (3) і (2),
(6) $B \to (C \to \neg(B \to \neg C))$ — теорема 155 f),
(7) $C \to \neg(B \to \neg C)$ — MP з (4) і (6),
(8) $\neg(B \to \neg C)$ — MP з (5) і (7).

За теоремою дедукції маємо $A \to B, A \to C \vdash A \to \neg(B \to \neg C) \leftrightarrow A \to \to (B \wedge C)$.

k) $\vdash (A \wedge B) \to A$

(Друге правило кон'юнкції (ПК2))

(1) $\neg A \to (B \to \neg A)$ — аксіома A1),
(2) $\neg(B \to \neg A) \to A$ — правило контрапозиції,
(3) $(A \wedge B) \to A$ — тотожність і закон комутативності для $\wedge$.

l) $A \to B, C \to B \vdash (A \vee C) \to B$

(Друге правило диз'юнкції (ПД2))

(1) $A \to B$ — гіпотеза,
(2) $C \to B$ — гіпотеза,
(3) $\neg B \to \neg A$ — правило контрапозиції,
(4) $\neg B \to \neg C$ — правило контрапозиції,
(5) $\neg B \to (\neg A \wedge \neg C)$ — теорема 155 i),
(6) $\neg(\neg A \wedge \neg C) \to B$ — правило контрапозиції,
(7) $(A \vee C) \to B$ — закони де Моргана.

За теоремою дедукції маємо $A \to B, C \to B \vdash (A \vee C) \to B$. ■

6.1.6. Несуперечність і повнота числення висловлювань

Визначення 145. *Теорія Th називається несуперечною, якщо або формула A, або формула ¬A є теоремою теорії Th, тобто коли існує доведення в Th або A, або ¬A.*

Теорема 156 (*Теорема про повноту*). *Формула A є теоремою ЧВ тоді і тільки тоді, коли A — тавтологія.*

Доведення. З одного боку, доведення випливає з того, що аксіоми є тавтологіями (у чому легко переконатися за допомогою таблиць істинності). А коли правило *MP* застосовується до тавтологій, то знову отримуємо тавтологію (див. приклад 6.1.3, задача 2)). Отже, довільна теорема ЧВ є тавтологією.

Щоб довести теорему в другий бік, скористаємося з такої леми:

Лема 7. *Нехай A є формулою, а B_1, B_2,..., B_k — пропозиційними змінними формули A, і нехай задана деяка інтерпретація h. Покладемо B'_i рівним B_i, якщо $h(B_i) = 1$, і B'_i дорівнює $\neg B_i$, якщо $h(B_i) = 0$; а також $A' = A$, якщо $h(A) = 1$, і $A' = \neg A$, якщо $h(A) = 0$. Тоді B'_1, B'_2, ..., $B'_k \vdash A'$.*

Доведення виконується методом математичної індукції за числом n входжень логічних зв'язок у формулу A (вважається, що формула A записана без скорочень). Якщо $n = 0$, то A є пропозиційною літерою B і лема 7 набуває вигляду: $B \vdash B$ або $\neg B \vdash \neg B$. Отже, для $n = 0$ лема 7 справедлива. Припустимо, що лема 7 справедлива для всіх $j < n$.

Випадок 1. A має вигляд заперечення $\neg B$. Число входжень логічних зв'язок у B, очевидно, менше від n.

Випадок 1*a*. Нехай $h(B) = 1$, тоді $h(A) = 0$. Отже, $B' = B$, а $A' = \neg A$. За припущенням індукції для формули B маємо $B'_1, B'_2,..., B'_k \vdash B$, а з теореми 155 *c*) і MP випливає, що $B'_1, B'_2,..., B'_k \vdash \neg\neg B$, а $\neg\neg B$ є формулою A'.

Випадок 1*b*. Нехай $h(B) = 0$, тоді $B' = \neg B$, а $A = A'$. За припущенням індукції маємо $B'_1, B'_2,..., B'_k \vdash \neg B$, що й потрібно було довести, оскільки $\neg B = A'$.

Випадок 2. Нехай A має вигляд $B \to C$. Тоді число входжень логічних зв'язок у B і C менше ніж в A. За припущенням індукції $B'_1, B'_2,..., B'_k \vdash B$, $B'_1, B'_2,..., B'_k \vdash C$.

Випадок 2*a*. Нехай $h(B) = 0$, тоді $h(A) = 1$ і $B' = \neg B$, а $A' = A$. За припущенням індукції маємо $B'_1, B'_2,..., B'_k \vdash \neg B$, а за теоремою 155*d*) отримуємо, що $B'_1, B'_2,..., B'_k \vdash B \to C$, де $B \to C$ є формулою A.

Випадок 2*b*. Нехай $h(C) = 1$ і $h(A) = 1$, тоді $C' = C$, а $A' = A$. За припущенням індукції маємо $B_1', B_2', ..., B_k' \vdash C$ і з аксіоми А1) отримуємо $B_1', B_2', ..., B_k' \vdash B \to C$, де $B \to C$ є формула A.

Випадок 2*c*. Нехай $h(B) = 1$ і $h(C) = 0$, тоді $h(A) = 0$ і $A' = \neg A$, $B' = B$, $C' = \neg C$. За припущенням індукції маємо $B_1', B_2', ..., B_k' \vdash B$, $B_1', B_2', ..., B_k' \vdash \neg C$, звідки за теоремою 155*f*) отримуємо $B_1', B_2', ..., B_k' \vdash \neg(B \to C)$, де $\neg(B \to C)$ є формулою A'. ■

Продовження доведення теореми 156. Нехай A є тавтологією, а формули $B_1', B_2', ..., B_k'$ — пропозиційними літерами формули A. Для довільної інтерпретації літер B_i за лемою 7, маємо $B_1', B_2', ..., B_k' \vdash A$, де $A' = A$, оскільки A є тавтологією. Тоді у випадку, коли B_k істинна, застосування леми 7 дає можливість отримати $B_1', B_2', ..., B_k' \vdash A$, а також і у випадку, коли B_k хибна, отримуємо $B_1', B_2', ..., \neg B_k' \vdash A$. За теоремою дедукції маємо $B_1', B_2', ..., B_{k-1}' \vdash B_k \to A$, $B_1', B_2', ..., B_{k-1}' \vdash \neg B_k \to A$. За теоремою 155 *g*) отримуємо $B_1', B_2', ..., B_{k-1}' \vdash A$. Нарешті, виключаючи всі B_i', через k кроків отримаємо $\vdash A$. ■

Доведена теорема дає можливість довести теорему, обернену до теореми дедукції.

Теорема 157. *Якщо* $\vdash A \to (B \to ... \to (C \to D))...)$, *то* $A, B, ..., C \vdash D$.

Доведення. Якщо $A \to (B \to ... \to (C \to D)...)$ є теоремою, то нехай $C_1, C_2, ..., C_k$ є її доведенням. Тоді доведення, що має вигляд

$C_1, C_2, ..., C_k$,

$A \to (B \to ... \to (C \to D)...)$,

A — гіпотеза,

$B \to (... \to (C \to D)...)$ — за правилом MP з попередніх двох формул,

B — гіпотеза,

... ...

$C \to$ D — за правилом MP з попередніх двох формул,

C — гіпотеза,

D — за правилом MP з попередніх двох формул

є доведенням формули D з $A, B, ..., C$, тобто $A, B, ..., C \vdash D$. ■

Наслідок 44. *Формула* D *є логічним наслідком формул* $A, B, ..., C$ *тоді і тільки тоді, коли* $A \wedge B \wedge ... \wedge C \to D$ *є тавтологією.*

Доведення. Якщо $A \wedge B \wedge ... \wedge C \to D$ тавтологія, то $A \wedge B \wedge ... \wedge C \to$ $\to D \leftrightarrow \neg(A \wedge B \wedge ... \wedge C) \vee D \leftrightarrow \neg A \vee \neg B \vee ... \vee \neg C \vee D \leftrightarrow$

$\leftrightarrow \neg A \vee \neg B \vee \ldots \vee (C \to D) \leftrightarrow \cdots \leftrightarrow A \to (B \to \ldots (C \to D)\ldots)$ теж є тавтологією. За теоремою 157 маємо $A, B, \ldots, C \vdash D$. ■

Наслідок 45. *Формула D є логічним наслідком формул A, B, ..., C тоді і тільки тоді, коли формула $A \wedge B \wedge \ldots \wedge C \wedge \neg D$ суперечність.*

Доведення пропонується як проста вправа.

Приклад 6.1.6. Довести, що $\neg P$ є логічним наслідком висловлювань $P \to Q$ і $\neg Q$.

Доведення. За наслідком 44, якщо формула $\neg P$ є наслідком, то формула $((P \to Q) \wedge \neg Q) \to \neg P$ має бути тавтологією. Скориставшись таблицями істинності, отримуємо:

P	Q	$\neg P$	$\neg Q$	$P \to Q$	$(P \to Q) \wedge \neg Q$	$((P \to Q) \wedge \neg Q) \to \neg P$
1	1	0	0	1	0	1
1	0	0	1	0	0	1
0	1	1	0	1	0	1
0	0	1	1	1	1	1

Отже, $\neg P$ є логічним наслідком висловлювань $P \to Q$ і $\neg Q$. ♠

Теорема 156 свідчить про те, що аксіоматизація числення висловлювань є правильною, оскільки за цією теоремою синтаксичний наслідок (наслідок, отриманий за допомогою формального доведення з аксіом) також є наслідком семантичним (тобто тавтологією), і навпаки.

Теорема 158. *Теорія ЧВ є несуперечною теорією.*

Доведення теореми є простим наслідком теореми 156. Дійсно, якщо A ($\neg A$) є теоремою ЧВ, то формула A ($\neg A$) має бути тавтологією. Але тоді формула $\neg A$ (A) не може бути тавтологією і за теоремою 156 не є теоремою ЧВ. ■

6.1.7. Числення висловлювань і булева алгебра

Нехай S означає множину всіх формул ЧВ, на яких визначені операції $\neg$, $\vee$, $\wedge$. За теоремою 152, пару $G = (S, \Omega = \{\neg, \vee, \wedge\})$ можна розглядати як універсальну алгебру. Визначимо на множині S відношення R таким чином:

$$ARB \Leftrightarrow A \to B.$$

Оскільки для довільних формул числення висловлювань маємо $\vdash A \to A$ і $A \to B, B \to C \vdash A \to C$, то звідси випливає, що відношення R є від-

ношенням рефлексивним і транзитивним, тобто відношення R є відношенням квазіпорядку. З властивостей відношення квазіпорядку випливає, що відношення еквівалентності R', яке відповідає R, таке, що

$$AR'B \Leftrightarrow ARB \wedge BRA \Leftrightarrow A \leftrightarrow B.$$

Це означає, що відношення R' є відношенням логічної еквівалентності.

Якщо h є деяка інтерпретація, що визначена на множині S, то з означення h випливає, що

$$h(A \leftrightarrow B) = h(A) = h(B),$$

а звідси маємо, що коли $h(A) = h(B)$, то $h(\neg A) = h(\neg B)$ і $h(A \wedge B) = h(A) \wedge h(B)$.

Іншими словами, відношення R' є відношенням конгруенції на множині S і за цим відношенням можна побудувати фактор алгебру $G/R' = (\{S\}, \Omega)$, де $\{S\}$ — множина класів еквівалентності. Алгебра G/R' називається **алгеброю Лінденбаума** або **алгеброю висловлювань.** Основна властивість цієї алгебри випливає з такої теореми.

Теорема 159. *Алгебра G/R ізоморфна булевій алгебрі* [36, 24].

Доведення. Поставимо у відповідність елементові 0 булевої алгебри формулу $A \wedge \neg A$, а елементові 1 — формулу $\neg(A \wedge \neg A)$. За законами де Моргана, булеву алгебру можна також трактувати як алгебру з операціями $\neg$ і $\wedge$. Для доведення необхідно показати справедливість усіх тотожностей булевої алгебри. Деталі цього показу пропонуються як вправи. ■

Приклад застосування логіки висловлювань

Нехай задано деякий опис фрагменту операційної системи:

A — «поява сигналу переривання, що призначається для процесу»,

P — «сигнал додається до множини сигналів, які чекають на виклик через процес»,

B — «зараз сигнал заблокований через процес»,

D — «процес відбирає сигнал»,

S — «поточний стан (контекст) процесу запам'ятовується»,

M — «визначається нова маска сигналу переривання»,

H — «наступає виклик процедури обслуговування сигналів переривання»,

N — «процедура обслуговування сигналів переривання викликана звичайним способом»,

R — «процес визначає виконання з попереднього контексту»,

I — «процес повинен сам відкрити попередній контекст».

З підручника про операційні системи довідуємося, що «$A \to P$, $(P \wedge \neg B) \to D$, $D \to (S \wedge M \wedge H)$, $(H \wedge N) \to R$, $(H \wedge \neg R) \to I$». Які наслідки можна одержати з цих припущень?

Зупинимося на питанні, коли формула $A \wedge \neg B \wedge \neg R$ буде істинна, тобто коли з'явиться сигнал переривання, який не залишиться заблокованим через процес, або що те саме, що процес не захоче відновити свого виконання в попереднім контексті?

Як наслідок, можемо отримати з формул $A \to P$ і $A \wedge \neg B$, що $P \wedge \neg B$. Далі, використовуючи припущення $(P \wedge \neg B) \to D$ і $D \to$ $\to (S \wedge M \wedge H)$, отримуємо $S \wedge M \wedge H$. Зокрема, істинною є формула H. За допомогою короткого доведення можна отримати як наслідок $\neg N$ з припущень $H \wedge \neg R$ і $(H \wedge N) \to R$. Оскільки $(H \wedge \neg R) \to I$, то з $H \wedge \neg R$ випливає $I \wedge \neg N$. Можна також показати, що коли формула A істинна, а формули B і R хибні, то формула I істинна, а тому формула N хибна, що означає, що процедура обслуговування сигналів переривання не викликається звичайним чином, у зв'язку з чим процес повинен відкрити свій попередній контекст. Разом з цим було також показано, що формули P, D, S, M і H істинні. ♠

6.2. Методи доведення тавтологій у ЧВ

У попередньому розділі було показано, як можна за допомогою таблиць істинності переконатися в тому, що дана формула є тавтологією чи суперечністю. У методі таблиць істинності потрібно розглядати всі можливі інтерпретації формули і тому цей метод часто називають **тривіальним.** Але його застосування вимагає великих затрат часу. Тому були розроблені й інші методи перевірки того, що дана формула є тавтологією. Розглянемо деякі з таких методів.

Метод Куайна. Метод Куайна є безпосереднім узагальненням тривіального методу. Нехай $\{p, q, ..., r\}$ упорядкована множина формул, які входять до запису формули $P(p, q, ..., r)$. Візьмемо першу з цих формул — p і припишемо їй, наприклад, значення 1 (0). Підставимо це значення у формулу P і виконаємо обчислення, які виникають після такої підстановки. Отримуємо нову формулу $P'(q, ..., r)$, до якої застосовуємо ту саму процедуру, тобто вибираємо формулу q, приписуємо їй значення 1 (0) і виконуємо обчислення. На якомусь кроці отримуємо, що формула P'' є тавтологією або суперечністю, незалежно від значень атомарних формул, які входять до складу формули P''. На цьому кроці алгоритм закінчує свою роботу і це означає, що в деяких випадках метод Куайна потребує розгляду меншої кількості інтерпретацій, а не всіх можливих інтерпретацій.

Приклад 6.2.1. Показати, що формула $P = (((p \wedge q) \to r) \wedge (p \to \to q)) \to (p \to r)$ є теоремою теорії ЧВ.

Розв'язок. Для доведення потрібно показати, що дана формула є тавтологією. Множина атомарних формул P включає формули $\{p, q, r\}$. Візьмемо формулу p і розглянемо можливі випадки:

a) $p = 1$, тоді $P = (((1 \wedge q) \to r) \wedge (1 \to q)) \to (1 \to r) = ((q \to r) \wedge q) \to \to r = P'$.

Тепер беремо q:

a1) $q = 1$, тоді $P' = ((1 \to r) \wedge 1) \to r = r \to r$ — тавтологія.

a2) $q = 0$, тоді $P' = ((0 \to r) \wedge 0) \to r = 0 \to r = 1$.

b) $p = 0$, тоді $P' = (((0 \wedge q) \to r) \wedge (0 \to q)) \to (0 \to r) = ((0 \to r) \wedge 1) \to \to 1 = 1 \to 1 = 1$.

Як бачимо, дана формула є тавтологією, а отже, є теоремою теорії ЧВ, причому інтерпретація формули r у формулі P несуттєва. ♠

Метод редукції. Цей метод дозволяє з'ясувати, є чи ні дана формула тавтологією шляхом «зведення до абсурду». Даний метод є ефективним у тому випадку, коли аналізована формула включає велику кількість імплікацій.

Нехай формула P має вигляд імплікації, наприклад, $P = Q \to R$. Припустимо, що для деякої інтерпретації h формула P хибна. Тоді, згідно з таблицею істинності для імплікації, маємо $h(Q) = 1$ і $h(R) = 0$. Тепер замість формули P будемо розглядати формули Q і R, застосовуючи той самий метод. У результаті, коли формула є тавтологією, то прийдемо до суперечності з тим, що деяка атомарна формула або деяка підформула за однієї і тієї самої інтерпретації має бути одночасно істинною і хибною, що неможливо.

Приклад 6.2.2. З'ясувати, чи є формула $P = ((p \wedge q) \to r) \to (p \to \to (q \to r))$ тавтологією.

Розв'язок. Припустимо, що існує інтерпретація h, у якій $h(P) = 0$. Тоді $h(p \to (q \to r)) = 0$ і $h((p \wedge q) \to r) = 1$. Застосовуючи ту саму процедуру до першої з формул, отримуємо: $h(p) = 1$ і $h(q \to r) = 0$. Звідси випливає, що $h(p) = 1$, $h(q) = 1$ і $h(r) = 0$. Отриманий результат суперечить тому, що $h((p \wedge q) \to r) = 1$ і ця суперечність свідчить про те, що формула P є тавтологією. ♠

Метод резолюцій. Для розгляду цього методу введемо деякі означення.

Літерою називається атомарна формула або заперечення атомарної формули. **Диз'юнктом** називається диз'юнкція літер. **Диз'юнктом одночленом** називається однолітеровий диз'юнкт. Якщо диз'юнкт не

містить літер, то він називається **пустим диз'юнктом** і позначається через 0. Літери L і $\neg L$ називаються **контрарними.**

Розглянемо диз'юнкти $C: P$ і $C': \neg P \vee Q$. Застосовуючи закони для диз'юнкції, із C і C' можна отримати диз'юнкт C'': $Q = P \vee (\neg P \vee Q) = (P \vee \neg P) \vee Q$. Діючи таким чином з довільною парою диз'юнктів, а не тільки з одночленними диз'юнктами, отримуємо **правило резолюцій.**

Формально це правило має таке означення. **Для довільних диз'юнктів C і C', якщо існує літера L у диз'юнкті C, контрарна літері L' у диз'юнкті C', то відрізаючи літери L і L' у диз'юнктах C і C' відповідно, будуємо диз'юнкцію решти елементів. Побудований таким чином диз'юнкт називається резольвентою диз'юнктів C і C'.**

Важливою властивістю резольвенти двох диз'юнктів C_1 і C_2 є те, що резольвента цих диз'юнктів є їх логічним наслідком, як випливає з теореми 155 h).

Визначення 146. *Нехай S деяка множина диз'юнктів. Резолюційним виведенням або резолюційним доведенням диз'юнкту C з множини S називається така скінченна послідовність диз'юнктів $C_1, C_2, \ldots, C_k$, що кожний диз'юнкт цієї послідовності C_i або належить до множини S, або є резольвентою диз'юнктів, які ідуть раніше C_i в цій послідовності, причому $C = C_k$.*

Виведення пустого диз'юнкту 0 із S називається **виведенням суперечності** S або **спростуванням** S, а сама множина S у такому випадку називається суперечною.

Говорять, що диз'юнкт C може бути отриманий із множини диз'юнктів S, якщо існує резолюційне виведення C із S.

Отже, для того щоб перевірити, є чи ні задана формула логічним наслідком деякої множини диз'юнктів, необхідно до цієї множини диз'юнктів додати заперечення даної формули і впевнитися в тому, що отримана множина є суперечною. У цьому і полягає суть методу резолюцій.

Правило резолюцій є дуже важливим правилом, оскільки воно застосовне не тільки до формул логіки висловлювань, а й до формул логіки предикатів першого порядку.

Приклад 6.2.3. 1) Чи є формула C логічним наслідком множини диз'юнктів $S' = \{P, \neg P \vee Q \vee R, \neg Q \vee C, \neg R \vee C\}$?

Розв'язок. Згідно із зазначеним вище, будуємо таку множину диз'юнктів $S = \{P, \neg P \vee Q \vee R, \neg Q \vee C, \neg R \vee C, \neg C\}$ і застосовуємо правило резолюцій до множини S:

(1) P — диз'юнкт 1,
(2) $\neg P \vee Q \vee R$ — диз'юнкт 2,
(3) $\neg Q \vee C$ — диз'юнкт 3,
(4) $\neg R \vee C$ — диз'юнкт 4,
(5) $\neg C$ — диз'юнкт 5,
(6) $Q \vee R$ — за ПР з (1) і (2),
(7) $\neg R$ — за ПР з (4) і (5),
(8) $\neg Q$ — за ПР з (3) і (5),
(9) Q — за ПР із (6) і (7),
(10) 0 — за ПР з (8) і (9).

Дана множина є суперечною, оскільки з неї виводиться пустий диз'юнкт. А це означає, що формула C є логічним наслідком множини диз'юнктів S'.

2) Показати суперечність множини диз'юнктів $S = \{P \vee Q, P \vee R, \neg Q \vee \neg R, \neg P\}$.

Розв'язок. Застосувавши правило резолюцій, отримуємо:

(1) $P \vee Q$ — диз'юнкт 1,
(2) $P \vee R$ — диз'юнкт 2,
(3) $\neg Q \vee \neg R$ — диз'юнкт 3,
(4) $\neg P$ — диз'юнкт 4,
(5) $P \vee \neg R$ — за ПР з (1) і (3),
(6) Q — за ПР з (1) і (4),
(7) $P \vee \neg Q$ — за ПР з (2) і (3),
(8) R — за ПР з (2) і (4),
(9) P — за ПР з (2) і (5),
(10) $\neg R$ — за ПР з (3) і (6),
(11) $\neg Q$ — за ПР з (3) і (8),
(12) $\neg R$ — за ПР з (4) і (5),
(13) $\neg Q$ — за ПР з (4) і (7),
(14) 0 — за ПР з (4) і (9).

Зауважимо, що доведення суперечності множини диз'юнктів з використанням правила резолюцій є недетермінованою процедурою. Справа в тім, що резольвенти двох диз'юнктів можна будувати різними способами. Даний приклад показує, що в ньому стратегія застосування правила резолюцій не була оптимальною. Виведення пустого диз'юнкту з множини S можна виконати, використовуючи меншу кількість резольвент, наприклад:

(5) Q — за ПР з (1) і (4),
(6) R — за ПР з (2) і (4),
(7) $\neg Q$ — за ПР з (3) і (6),
(8) 0 — за ПР з (5) і (7). ♠

Можливість будувати різної довжини резолюційні виведення є наслідком недермінованості застосування правила резолюцій. Ця недетермінованість у застосуванні правила резолюцій є недоліком даного методу.

Метод семантичного табло. Метод семантичного табло (СТ) є одним з ефективних алгоритмів перевірки виконуваності формул числення висловлювань. Цей метод може бути використаний (і використовується) як загальний алгоритм не тільки для числення висловлювань, а й для інших логік (модальної, темпоральної, предикатів). Головна особливість цього методу для числення висловлювань є досить проста — **пошук моделі**, на якій дана формула виконується. Продемонструємо спочатку цей метод на прикладі.

Приклад застосування методу СТ

а) Дана формула $A = p \wedge (\neg q \vee \neg p)$. Чи існує хоча б одна інтерпретація h, при якій A виконується? Очевидно, що

$$h(A) = 1 \text{ тоді і тільки тоді, коли } h(p) = 1 \text{ і } h(\neg q \vee \neg p) = 1.$$

Звідси випливає, що $h(A) = 1$ тоді і тільки тоді, коли

$$\text{a) } h(p) = 1 \text{ і } h(\neg q) = 1 \text{ або } h(p) = 1 \text{ і } h(q) = 0;$$

$$\text{b) } h(p) = 1 \; h(\neg p) = 1 \text{ або } h(p) = 1 \text{ і } h(p) = 0$$

Це означає, що A виконується тоді і тільки тоді, коли існує несуперечна інтерпретація — інтерпретація, яка не включає контрарних пар літер. Зазначимо, що p, q є формулами атомарними і питання про виконуваність формули A зводиться до питання про виконуваність її атомарних формул. Легко переконатися, що множина атомарних формул виконується тоді і тільки тоді, коли ця множина не включає контрарних літер. У даному прикладі множина літер, які відповідають першій інтепретації, не має контрарних пар. Це означає, що ця множина виконується при такій інтерпретації:

$$h(p) = 1, \quad h(q) = 0.$$

Множина, що відповідає другій інтерпретації, включає контрарну пару і не генерує інтерпретації для формули A, оскільки одна й та сама атомарна формула не може бути одночасно істинною і хибною при одній і тій самій інтерпретації.

b) Розглянемо ще один приклад. Чи виконується формула

$$A = (p \wedge q) \wedge (\neg p \vee \neg q)?$$

Якщо h деяка інтерпретація для A, то

$h(A) = 1$ тоді і тільки тоді, коли $h(p \wedge q) = 1$ і $h(\neg p \vee \neg q) = 1$.

Звідси випливає, що $h(A) = 1$ тоді і тільки тоді, коли

a) $h(p) = h(q) = 1$ і $h(p) = 0$;
b) $h(p) = h(q) = 1$ і $h(q) = 0$.

Множина літер, що відповідає першій інтерпретації, є такою: $\{p, q, \neg p\}$, а множина літер для другої інтерпретації — $\{p, q, \neg q\}$. Обидві множини включають контрарні пари літер і для формули A не існує інтерпретації, при якій ця формула виконується. ♠

Для побудови всіх множин формул і інтерпретацій, які відповідають цим формулам, скористаємося з позначених дерев (див. розділ 5.3.8) [15].

Коренем такого дерева є вершина, позначена формулою A, а внутрішні вершини дерева позначені множинами формул, які виникають у процесі побудови СТ. Листкам дерева відповідають множини літер, які і є інтерпретаціями. Крім того, якщо листок дерева, якому відповідає певна множина літер, включає контрарні літери, то позначаємо його знаком «×», у протилежному випадку — знаком ∘. Отримане в такий спосіб дерево будемо називати **семантичним табло (СТ)**.

Для формул з попереднього прикладу СТ має такий вигляд:

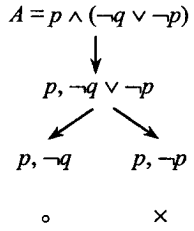


Рис. 6.2.1. СТ для формули $A = p \wedge (\neg q \vee \neg p)$

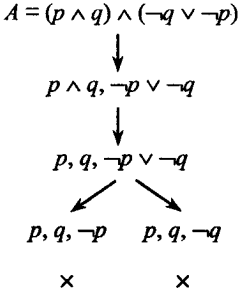


Рис. 6.2.2. СТ для формули $A = (p \wedge q) \wedge (\neg p \vee \neg q)$

При побудові СТ всі формули числення висловлювань поділяють на два класи:

— α-формули для кон'юнкції, що виконуються тоді і тільки тоді, коли виконуються обидва аргументи цієї кон'юнкції;

— β-формули для диз'юнкції, що виконуються тоді і тільки тоді, коли принаймні один з аргументів цієї диз'юнкції виконується.

Правила перетворення для α- і β-формул, що застосовуються при побудові СТ, записують у вигляді таблиць.

Таблиця 6.2.1

ТАБЛИЦЯ ПРАВИЛ ДЛЯ α-ФОРМУЛ

α	α_1	α_2
$\neg\neg A$	A	
$A_1 \wedge A_2$	A_1	A_2
$\neg(A_1 \vee A_2)$	$\neg A_1$	$\neg A_2$
$\neg(A_1 \to A_2)$	A_1	$\neg A_2$
$A_1 \leftrightarrow A_2$	$A_1 \to A_2$	$A_2 \to A_1$

Таблиця 6.2.2

ТАБЛИЦЯ ПРАВИЛ ДЛЯ β-ФОРМУЛ

β	β_1	β_2
$B_1 \vee B_2$	B_1	B_2
$\neg(B_1 \wedge B_2)$	$\neg B_1$	$\neg B_2$
$B_1 \to B_2$	$\neg B_1$	B_2
$\neg(B_1 \leftrightarrow B_2)$	$\neg(B_1 \to B_2)$	$\neg(B_2 \to B_1)$

Тепер розглянемо алгоритм побудови СТ для формул числення висловлювань.

АЛГОРИТМ ПОБУДОВИ СТ (*A*)

Вхід. Кожна вершина табло T позначається множиною формул. Спочатку T складається лише з однієї вершини — кореня цього дерева, який позначений формулою A.

Вихід. Дерево T, в якому всі вершини позначені множинами формул, а листки — символами $\times$ і $\circ$.

Метод.

Крок 1. Взяти листок l, який ще не розглядався і який позначений множиною формул $U(l)$.

Крок 2. Якщо $U(l)$ є множиною літер, то якщо ця множина включає контрарні пари, то позначаємо цю вершину знаком «$\times$», у протилежному випадку «$\circ$».

Крок 3. Якщо $U(l)$ не є множиною літер, то беремо формулу B з множини $U(l)$.

a) Якщо формула B є α-формулою, то будуємо нову вершину l' як сина вершини l і позначаємо її множиною формул

$$U(l') = (U(l)\backslash\{\{B\}\}) \cup \{\alpha_1, \alpha_2\}$$

(якщо B є формулою вигляду $\neg\neg B$, то член α_2 відсутній).

b) Якщо формула B є β-формулою, то будуємо дві вершини l і l' як синів вершини l і позначаємо їх відповідно множинами формул

$$U(l') = (U(l) \setminus B) \cup \{\beta_1\}$$
$$U(l'') = (U(l) \setminus B) \cup \{\beta_2\}$$

Крок 4. Якщо всі листки в дереві T мають позначки «$\times$» або «$\circ$», то слід закінчити роботу, інакше — перейти до виконання кроку 1.

Кінець побудови

Визначення 147. *СТ називається повним, якщо його побудова закінчена. Повне СТ називається* **замкнутим**, *якщо всі його листки позначені символом* «$\times$», *інакше воно називається* ***відкритим*** (*тобто коли існує принаймні один листок, позначений символом* «$\circ$»).

Теорема 160. *Побудова СТ завжди є скінченною процедурою.*

Доведення. Нехай дерево T є СТ для формули A. Припустимо, що формула A не включає зв'язки $\leftrightarrow$. Для кожного листка $l \in T$ нехай $b(l)$ є числом бінарних зв'язок із множини $U(l)$ і нехай $n(l)$ є числом операцій заперечення в $U(l)$. На цій основі визначимо число

$$W(l) = 2b(l) + n(l).$$

На кожному кроці побудови дерева T додаємо або одну нову вершину l або дві нові вершини l і l' такі, що $W(l') < W(l)$ і $W(l'') < W(l)$. Наприклад, якщо застосовуємо α-правило до формули $\neg(A_1 \vee A_2)$, то отримуємо $\neg A_1$ і $\neg A_2$, звідки маємо

$$W(l') = k + 0 + 2 < k + 2 + 1 = W(l),$$

де k є сумарним числом зв'язок в A_1 і A_2. Оскільки $W(l) \geq 0$, то дерево T не може бути нескінченним.

Якщо формула A включає зв'язки $\leftrightarrow$, то спочатку вилучимо їх із формули A шляхом застосування тотожності

$$A' \leftrightarrow B' \Leftrightarrow (A' \rightarrow B') \wedge (B' \rightarrow A'),$$

а потім застосуємо вищенаведені викладки до отриманої формули. Подальші деталі цього доведення пропонуються як вправи. ■

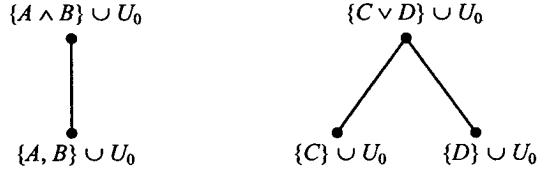
Теорема 161. *Нехай дерево T є повним СТ для формули A. Формула A не виконується тоді і тільки тоді, коли СТ замкнуте.*

Доведення. Покажемо, що коли довільне піддерево дерева T з коренем у вершині n є замкнутим, то множина формул $U(n)$ не є виконуваною.

Доведення виконується методом математичної індукції за висотою h вершини n у дереві T.

Базис індукції. Якщо $h = 0$, то n є листком. Оскільки дерево T є замкнутим, то $U(n)$ включає принаймні одну пару контрарних літер $\{p, \neg p\}$. А це означає, що $U(n)$ не виконується.

Крок індукції. Якщо $h > 0$, то звідси випливає, що при побудові сина s вершини n було застосоване деяке α- або β-правило, що ілюструють нижченаведені рисунки.



Випадок 1. Було застосоване α-правило. Тоді $U(n) = \{A \wedge B\} \cup U_0$ і $U(s) = \{A, B\} \cup U_0$ для деякої множини формул U_0. Тоді в силу того, що висота вершини s дорівнює $h - 1$ і СТ для вершини s замкнуте, то згідно з припущенням індукції множина формул $U(s)$ не є виконуваною. Нехай v є довільною інтерпретацією. Оскільки $U(s)$ не є виконуваною, то $v(A') = 0$ для деякої формули $A' \in U(s)$. Тоді можливі такі випадки:

a) для деякої формули $A_0 \in U_0$ і $v(A_0) = 0$; тоді $A_0 \in U_0 \subseteq U(s)$;

b) $v(A) = 0$ і тоді $v(A \wedge B) = 0$;

c) $v(B) = 0$ і тоді $v(A \wedge B) = 0$.

Як бачимо, у кожнім з випадків $v(A) = 0$ для деякої формули $A \in U(n)$, а це означає, що $U(n)$ не є виконуваною.

Випадок 2. Було застосоване β-правило. Тоді $U(n) = \{C \vee D\} \cup U_0$ і $U(s') = \{B\} \cup U_0$, $U(s'') = \{C\} \cup U_0$. За припущенням індукції отримуємо, що $U(s')$ і $U(s'')$ не виконуються. Нехай v є довільною інтерпретацією. Тоді можливі такі випадки:

a) $v(A_0) = 0$ для деякої формули $A_0 \in U_0 \subseteq U(s')$;

b) якщо $v(A_0) = 1$, то оскільки як $U(s')$, так і $U(s'')$ не є виконуваними множинами формул, маємо $v(C) = v(D) = 0$.

В обох випадках отримуємо, що множина формул $U(n)$ не виконується.

Покажемо тепер повноту методу СТ, тобто якщо формула A не виконується, то СТ для A замкнуте. З цією метою скористаємося правилом контрапозиції, за яким потрібно довести таке твердження: якщо СТ для формули A відкрите, то формула A виконується.

Доведення методом математичної індукції за висотою вершини n у дереві T семантичного табло для формули A.

Базис індукції. Якщо $h = 0$, то A є атомарною формулою, отже, $A = p$ або $A = \neg p$, де p є атомарною формулою. Оскільки СТ для A складається з єдиної вершини 0, позначеної множиною $U(0) = \{p\}$ або $U(0) = \{\neg p\}$, то СТ є відкритим і формула A є виконуваною при інтерпретації $v(p) = 1$ або при інтерпретації $v(p) = 0$.

Крок індукції. Нехай теорема має місце для $h < n$ і СТ для A має висоту n.

Якщо A є α-формула, наприклад, $A = A_1 \wedge A_2$, то вершина 0, з позначкою $U(A)$, має єдиного сина 1, з позначкою у вигляді множини $U(l) = \{A_1, A_2\}$, причому висота вершини є меншою, ніж висота n. Оскільки СТ для A є відкритим, то СТ для $\{A_1, A_2\}$ теж є відкритим і за припущенням індукції формули A_1 і A_2 є виконуваними, а тоді і формула A теж є виконуваною.

Якщо A є β-формула, наприклад $A = A_1 \vee A_2$, то оскільки СТ для A є відкритим, то тоді СТ для A_1 або для A_2 теж є відкритим. За припущенням індукції це означає, що формула A_1 або A_2 є виконуваною. Але тоді і формула A теж є виконуваною. ■

Наслідок 46. *Формула* A *виконується тоді і тільки тоді, коли дерево T відкрите.*

Доведення випливає безпосередньо з теореми 161 і закону контрапозиції. ■

Наслідок 47. *Формула A є тавтологією тоді і тільки тоді, коли СТ для $\neg A$ замкнуте.*

Доведення випливає з теореми 161 і його побудова пропонується як вправа. ■

Наслідок 48. *Метод СТ є процедурою перевірки виконуваності для формул числення висловлювань.*

Доведення. Нехай формула A є теоремою логіки висловлювань. З теореми 160 випливає, що побудова СТ T для формули A є процедурою скінченною і після закінчення цієї побудови табло T є повним. З наслідку 46 випливає, що A виконується тоді і тільки тоді, коли повне табло T є відкритим. ■

6.3. Контрольні питання, задачі і вправи

1. Що називається атомарною формулою?
2. Яка формула називається ППФ?
3. Коли формальна логічна теорія вважається визначеною?
4. Яка формула називається а) тавтологією, b) суперечністю?
5. Скільки аксіом і правил виведення має числення висловлювань?
6. Назвіть аксіоми числення висловлювань. Які існують інші аксіоматизації числення висловлювань?
7. Скільки правил виведення розглядалося в численні висловлювань?
8. Яка теорія називається несуперечною?
9. Сформулювати теорему: а) дедукції, b) обернену до теореми дедукції.
10. Що називається літерою?
11. Які літери називаються контрарними?
12. Що називається диз'юнктом?
13. Що собою являє метод Куайна, резолюцій?
14. Назвати правила виведення числення висловлювань, які відрізняються від правила modus ponens.
15. Чи є теоремою числення висловлювань формула $A \vee \neg A$?
16. Чи є наведені нижче вирази ППФ численнями висловлювань:
    a) $(A \wedge B) \subseteq D$,
    b) $(A \wedge B) \to C$,
    c) $((A \to B) \wedge \neg B$,
    d) $(((\neg A) \to B) \to \neg(C \vee D))$.
17. Скількома способами можна розставити дужки в наведених нижче виразах:
    a) $A \to B \vee \neg B \wedge C$,
    b) $A \to B \to C \to \neg A \to \neg B$,
    c) $A \wedge B \vee C \wedge D \wedge C \vee A$?

18. Виписати всі підформули для таких формул:
a) $(((A \to B) \wedge (B \to C)) \to (\neg A \vee C))$,
b) $((A \to B) \to ((A \to \neg B) \to \neg B))$.
19. Побудувати таблиці істинності для формул:
a) $((P \to Q) \vee (P \to (Q \wedge P)))$,
b) $(\neg P \to \neg(Q \wedge P)) \to (P \vee R))$,
c) $((P \wedge (Q \to P)) \to \neg P)$,
d) $(((P \wedge \neg Q) \to Q) \to (P \to Q))$,
e) $((P \to (Q \vee R)) \to ((P \to Q) \to (P \to R)))$,
f) $((P \wedge (Q \vee \neg P)) \wedge ((\neg Q \to P) \vee Q))$.
20. Довести, що існує інтерпретація, при якій такі формули є істинними:
a) $\neg(P \to \neg P)$,
b) $((P \to Q) \to (Q \to P))$,
c) $((Q \to (P \wedge R)) \wedge \neg((P \vee R) \to Q))$.
21. Довести, що наведені нижче формули є тавтологіями:
a) $((P \to Q) \vee (Q \to P))$,
b) $((P \to Q) \vee (P \to \neg Q))$,
c) $(P \to (Q \to (P \wedge Q)))$,
d) $((P \to Q) \to ((Q \to R) \to (P \to R)))$,
e) $((\neg P \to \neg Q) \to (Q \to P))$,
f) $(P \to (Q \to P))$,
g) $((P \to Q) \to ((P \to (Q \to R)) \to (P \to R)))$,
h) $(P \to Q) \to P$,
i) $P \to (P \vee Q)$,
j) $P \vee \neg P$,
k) $(P \vee P) \to P$,
l) $Q \to (P \vee Q)$,
m) $(\neg P \to (P \to Q))$.
22. Для яких логічних значень змінних x, y, z, u, v, w наведені нижче вирази є хибними:
a) $(((x \to (y \wedge z)) \to (\neg y \to \neg x)) \to y)$,
b) $((x \wedge y) \vee (x \wedge z) \vee (y \wedge z) \vee (\neg x \wedge \neg z)$,
c) $(((x \vee y) \vee z) \to ((x \vee y) \wedge (x \vee z)))$,
d) $(((x \vee y) \wedge ((y \vee z) \wedge (z \vee x))) \to ((x \wedge y) \wedge z))$,
e) $((x \vee y) \to ((\neg x \wedge y) \vee (x \wedge \neg y)))$?
23. Довести, що
a) $\neg A \to B, A \to C, B \to D \vdash \neg C \to D$,
b) $A \to B, C \to \neg B \vdash A \to \neg C$,
c) $A \to P, A \wedge \neg B \vdash P \wedge B$,
d) $H \wedge \neg R, (H \wedge N) \to R \vdash \neg N$.

24. Довести, що формули RA1) — RA3) в аксіоматиці Россера і формули HAA1)—HAA4) в аксіоматиці Гільберта-Аккермана є теоремами числення висловлювань, тобто існує доведення з аксіом A1)—A3) за допомогою правила MP.

25. Побудувати СТ для формул із вправ 20 і 21 і переконатися в тому, що існує інтерпретація, при якій ці формули виконуються.

26. Розглянемо такі гіпотези. Якщо я поїду автобусом або метро, то спізнюся на побачення. Якщо поїду на таксі, то не спізнюся на побачення, але збанкрутую. Які з наступних висловлювань є наслідками даних гіпотез, тобто можуть бути формально доведені при даних гіпотезах?

a) Поїду на таксі.
b) Збанкрутую.
c) Не поїду метро.
d) Якщо поїду на таксі, то збанкрутую.
e) Якщо поїду автобусом, то не збанкрутую.

27. Нехай

A = «є дорослим»,
B = «є здоровим і сильним»,
Y = «роки минають»,
L = «життя важке»,
N = «ніхто мене не любить».

Які з наступних висловлювань є правильними наслідками? Обґрунтувати кожну відповідь.

a) Є дорослим тоді і тільки тоді, коли є здоровим і сильним.
b) Якщо я здоровий і сильний, то хтось мене любить.
c) Або літа минають, або є дорослим.
d) Життя важке, і мене ніхто не любить.

28. Зробила це Аня або Галя. Галя не могла одночасно читати і це зробити. Галя читала. Хто це зробив? Відповідь обґрунтувати за допомогою формального доведення із змінними *A*, *G*, *Z*.

29. Записати за допомогою логічної символіки наведені нижче висловлювання, вживаючи змінні для висловлювань. Подати формальні доведення отриманих формул.

а) Я піду додому (H) або залишусь тут і послухаю музику (S). Я не піду додому. Отже, я залишусь тут і послухаю музику.

б) Якщо Джон ляже спати сьогодні пізно (S), він буде вранці не у формі (D). Якщо він ляже спати не пізно, то йому буде здаватися, що не варто жити (L). Отже, або Джон буде завтра не в формі або йому здаватиметься, що не варто жити.

в) Заробітна плата зросте (W) тільки, якщо буде інфляція (J). Якщо буде інфляція, то збільшиться вартість життя (C). Заробітна плата зросте. Отже, збільшиться вартість життя.

г) Якщо 2 — просте число (P), то це найменше просте число (L). Якщо 2 — найменше просте число, то 1 не є простим числом (N). Число 1 не є простим числом. Отже, 2 — просте число.

д) Джон або перевтомився (E), або хворий (S). Якщо він перевтомився, то він дратується (C). Він не дратується. Отже, він хворий.

е) Якщо я поїду автобусом (B), а автобус запізниться (L), то я пропущу важливе побачення (M). Якщо я пропущу важливе побачення і почну засмучуватись (D), то мені не слід їхати додому (H). Якщо я не отримаю цієї роботи (I), то я почну засмучуватись і мені треба поїхати додому. Отже, якщо я поїду автобусом і автобус запізниться, то я отримаю цю роботу.

ж) Якщо 6 — складене число (S), то 12 — складене число (W). Якщо 12 — складене число, то існує просте число, яке більше ніж 12 (P). Якщо існує просте число, більше ніж 12, то існує складене число, більше ніж 12 (C). Якщо 6 ділиться на 2 (D), то 6 — складене число. Число 12 складене. Отже, 6 — складене число.

з) Якщо завтра буде холодно (C), то я одягну тепле пальто (H), якщо рукав буде полагоджений (S). Якщо завтра буде холодно, а рукав не буде полагоджений, тож я не одягну тепле пальто.

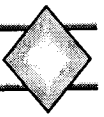
30. Якщо я тут, то сьогодні повинна бути п'ятниця. Якщо я не помилився, то сьогодні субота. Або сьогодні, то не вчора, або сьогодні п'ятниця. Не можу я помилитись, якщо я тут. П'ятниця, то не субота. Якщо сьогодні субота, то вчора була п'ятниця.

a) Чи є я тут?

b) Припустимо, що я не помилився. Чи можна з того отримати як наслідок, що вчора, то не сьогодні?

c) Якщо я не помилився, то який день був учора?

7 ЛОГІКИ НЕКЛАСИЧНІ



Числення висловлювань і числення предикатів першого порядку (яке буде розглянуто в наступному розділі) називаються **арістотелевськими логіками** або **класичними логіками.** Крім цих двох класичних логік існує цілий ряд логік, які називаються **некласичними.** Існування цих логік пояснюється тим, що не всі висловлювання можна записати за допомогою формул логік класичних. Зокрема, числення висловлювань має досить обмежені виразні можливості, тобто за допомогою формул цього числення можна виразити досить обмежений клас властивостей оточуючої нас дійсності. Наприклад, істинність формули «зараз є день» залежить від точки на земній кулі, бо якщо зараз у Києві 12-та година дня, то ця формула буде істинною, а, скажімо, у Вашингтоні це буде 1-ша година ночі і там ця формула буде хибною. Оскільки формули ЧВ ніяк не пов'язуються з якими-небудь зовнішніми факторами, то виразити наведене висловлювання засобами ЧВ неможливо. Отже, існують висловлювання, які вимагають інших, більш потужних формалізмів, ніж формалізм ЧВ. У зв'язку з цим було впроваджено і досліджено цілі класи логік, які дістали назву некласичних логік.

Некласичні логіки поділяють на два класи: перший клас включає логіки, які є розширенням класичних логік, а другий клас включає логіки, які є альтернативними до класичних логік. До першого класу, наприклад, відноситься **модальна логіка** і її різновиди — **темпоральна** і **динамічна** логіки. До другого класу відноситься **багатозначна логіка, нечітка логіка, інтуіціоністська логіка** тощо.

Розглянемо один з підкласів першого класу некласичних логік, елементи якого називають **пропозиційними модальними логіками**.

7.1. Пропозиційна модальна логіка (ПМЛ)

Синтаксис. Формули ПМЛ будуються за допомогою тих самих правил, які вживалися для побудови логіки висловлювань. Але ПМЛ додатково використовує два нових символи $\Box$ і $\Diamond$, які називаються

відповідно **модальними операторами загальності** і **існування.** Обидва ці оператори є унарними і діють на множині формул. Формули ПМЛ отримуємо за таким індуктивним означенням.

Визначення 148. **(синтаксис ПМЛ)**
1. Усі атомарні висловлювання є формулами ПМЛ.
2. Якщо A і B є формулами ПМЛ, то $\neg A$, $A \wedge B$, $A \vee B$, $A \to B$, $A \leftrightarrow B$, $\Box A$, $\Diamond A$ теж є формулами ПМЛ.
3. Решта слів, тобто слів, що збудовані не за правилами 1—2, *не є формулами ПМЛ.*

Оператори $\Box$ і $\Diamond$ пов'язані між собою за допомогою відношення двоїстості: $\Diamond A = \neg\Box\neg A$. Це відношення часто трактується як означення оператора $\Diamond$ за допомогою оператора $\Box$. Отже, кожну формулу ПМЛ можна записати користуючись лише оператором $\Box$.

Оператори $\Box$ і $\Diamond$ можуть читатися різними способами. Деякі зі способів читання цих операторів представлені нижче:

Оператор $\Box A$:
Необхідно, щоб була істинна A,
Завжди буде істинна A,
Вимагається, щоб була істинна A,
Покладаємо, що істинна A,
Відомо, що істинна A,
Довільне виконання програми дає результат, що задовольняє A.

Оператор $\Diamond A$:
Можливо, що істинна A,
Інколи буде істинна A,
Дозволяється, щоб була істинна A,
Не покладається супротивне до A,
Не відомо супротивне до A,
Існує таке виконання програми, яке дає результат, що задовольняє A.

Семантика. Для того, щоб визначити семантику формул ПМЛ, необхідно розглянути поняття **структури.**

Визначення 149. *Структурою називається пара (W, R), де W деяка непуста множина, а $R \subseteq W \times W$ бінарне відношення у множині W. Елементи множини W називаються* **точками.**

Визначення 150. *Нехай P є множиною атомарних формул ПМЛ. P-моделлю на структурі $F = (W, R)$ називається трійка $M = (W, R, f)$, де $f: P \to B(W)$ функція із множини формул P в булеан множини W. Для $p \in P$ множину $f(p)$ можна неформально інтерпретувати як множину точок з W, в яких висловлювання p є істинне.*

Якщо P є фіксованою множиною формул, то символ P у означенні моделі опускається і в цьому випадку говорять, просто **модель**.

Нехай $w \in W$ і A є формулою ПМЛ. Вираз $M \models_w A$ означає, що формула A є істинна в точці w моделі $M = (W, R, f)$.

Визначення 151. *ПМЛ-формула A називається* **виконуваною** *тоді і тільки тоді, коли існує модель M і деяка точка w цієї моделі така, що $M \models_w A$.*

Визначення 152 **(семантика формул ПМЛ).** *Семантика формул ПМЛ визначається на моделі $M = (W, R, f)$ таким чином:*

1. $M \not\models_w 0$.
2. $M \models_w A$, *якщо* $A \in P$ *і* $w \in f(A)$.
3. $M \models_w A \to B$, *якщо з* $M \models_w A$ *випливає* $M \models_w B$.
4. $M \models_w \Box A$, *якщо для кожного* $t \in W$ *такого, що* wRt *випливає* $M \models_t A$.

З останнього правила отримуємо, що формула $\Box A$ має бути істинна в усіх точках t моделі M, для яких відношення wRt правильне. Правила 1—4 називаються **базовими правилами**. За допомогою цих правил можна знайти значення формул: **1**, $\neg A$, $A \vee B$, $A \wedge B$, $A \leftrightarrow B$ і $\Diamond A$. Наприклад, сталу **1** можна записати як $0 \to A$, а формулу $\neg A$ як $A \to 0$. З правила $\neg A = A \to 0$ випливає таке семантичне правило:

1. $M \models_w \neg A$, якщо $M \not\models_w A$.
2. $M \models_w \Diamond A$, якщо $M \models_t A$ принаймні для одного $t \in W$ такого, що wRt.
3. $M \models_w A \vee B$, якщо $M \models_w A$ або $M \models_w B$.
4. $M \models_w A \wedge B$, якщо $M \models_w A$ і $M \models_w B$.

Визначення 153. *Формула A називається* **істинною** *в моделі M, якщо вона істинна в кожній точці цієї моделі, тобто $(\forall w \in W)M \models_w A$. Записується це так: $M \models A$.*

Формула A називається **істинною в структурі** *$F = (W, R)$, якщо вона істинна в кожній моделі $M = (W, R, f)$, тобто $(\forall M = (W, R, f))M \models A$.*

Формула A називається **модальною тавтологією**, *якщо вона істинна в усіх структурах (W, R). Записується це так: $\models A$.*

Теорема 162 *Наведені нижче формули є модальними тавтологіями:*

a) $\Box A \leftrightarrow \neg\Diamond\neg A$;
b) $\Box (A \to B) \to (\Box A \to \Box B)$;
c) $\Diamond(A \to B) \to (\Diamond A \to \Diamond B)$.

Доведення. а) Припустимо, що формула $\Box A$ істинна в моделі $M = (W, R, f)$ структури $F = (W, R)$. Відповідно до означення, для деякої точки $s \in W$ маємо $M \models_s \Box A$ і тоді формула $\Diamond\neg A$ є хибною в точці s, оскільки з того, що $\Box A$ є істинною в точці s, випливає, що A є істинною в усіх точках $s' \in W$ таких, що $(s, s') \in R$. А це означає, що $\neg\Diamond\neg A$ є істинною в усіх таких точках s'. Звідси випливає, що формула $\neg\Diamond\neg A$ істинна в точці s і в силу того, що модель M і структура F є довільними, то має місце така імплікація

$$\Box A \to \neg\Diamond\neg A.$$

Доведення оберненої імплікації подібне до даного і це доведення пропонується виконати як вправу.

b) Припустимо, що дане твердження хибне. Це може бути тоді і тільки тоді, коли формула $\Box(A \to B)$ істинна, а формула $\Box A \to \Box B$ хибна. Тоді $\Box A$ повинна бути істинна, а $\Box B$ хибна. Істинність формули $\Box A$ означає, що формула A істинна в усіх точках моделі. Істинність формули $\Box(A \to B)$ означає істинність формули $A \to B$ в усіх точках тієї самої моделі. Але з істинності попередньої формули випливає істинність формули B в усіх точках моделі, а це суперечить тому, що $\Box B$ є хибною.

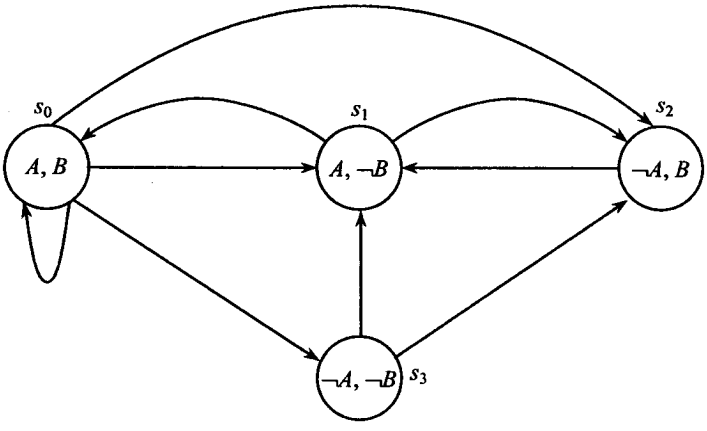
c) Припустимо, що дана формула хибна. Це може бути тоді і тільки тоді, коли $\Diamond(A \to B)$ істинна, а $(\Diamond A \to \Diamond B)$ хибна. Це означає, що $\Diamond(A \to B)$ істинна, $\Diamond A$ істинна, а $\Diamond B$ хибна. Істинність формул $\Diamond(A \to B)$ і $\Diamond A$ означає, що $\forall M = (W, R, f)\ \exists t, t' \in W$ такі, що $M \models_t A \to B$ і $M \models_{t'} A$.

Припустимо, що $M \not\models_{t'} B$. Звідси випливає, що $M \models_{t'} (\Diamond A \to \Diamond B)$ і $M \not\models_t \Diamond(A \to B)$, але тоді вся формула $\Diamond(A \to B) \to (\Diamond A \to \Diamond B)$ істинна в точці t' моделі M. Отримуємо суперечність з припущенням, що ця формула хибна. Ця суперечність доводить, що формула $\Diamond(A \to \to B) \to (\Diamond A \to \Diamond B)$ є модальною тавтологією. ■

Потрібно зауважити, що з пункту а) цього твердження випливає справедливість означення $\Box A \leftrightarrow \neg\Diamond\neg A$, яке було назване правилом двоїстості для модальних операторів $\Box$ і $\Diamond$.

Описана семантика формул ПМЛ називається **реляційною семантикою** або **семантикою Кріпке**, за іменем автора, який уперше ввів цю семантику до розгляду. Покажемо на прикладі як істинність чи хибність деякої формули ПМЛ у моделі Кріпке.

Приклад 7.1.1. Нехай дано формулу $\Box A \vee \Box B$ і модель Кріпке вигляду



Для цієї моделі маємо $W = \{s_0, s_1, s_2, s_3\}$ і

$$R = \{(s_0, s_0), (s_0, s_1), (s_0, s_2), (s_0, s_3), (s_1, s_0), (s_1, s_2), (s_2, s_1), (s_3, s_2), (s_3, s_1)\}.$$

Позначки станів показують, що в стані

- s_0 істинні формули A, B;
- s_1 істинні формули A, $\neg B$;
- s_2 істинні формули $\neg A$, B;
- s_3 істинні формули $\neg A$, $\neg B$.

У цій інтерпретації формула $\Box A \vee \Box B$ є

- хибна в стані s_0, оскільки стани s_0, s_1, s_2, s_3 є досяжними із стану s_0, і в стані s_3 формули A і B хибні;
- істинна в стані s_1, оскільки стани s_0 і s_2 є досяжними із стану s_1, і формула B є істинна в станах s_0 і s_2;
- істинна в стані s_2, оскільки стан s_1 є досяжним із стану s_2, і в стані s_1 істинна формула A;
- істинна в стані s_3, оскільки стани s_1, s_2 досяжні із стану s_3, і в стані s_1 істинна формула A, а в стані s_2 істинна формула B. ♠

Аксіоматична система для ПМЛ. Як уже зазначалося, ПМЛ є розширенням класичної логіки висловлювань. Формальна аксіоматична система для ПМЛ будується за тими самими правилами, що й для логіки висловлювань.

Формальна аксіоматична теорія для ПМЛ визначається таким чином:

1. Логічними символами ПМЛ є $\neg$, $\rightarrow$, (,) а також літери A_i алфавіту Al. Символи $\neg$, $\rightarrow$ називаються **пропозиційними зв'язками**, а символи A_i — **пропозиційними змінними** або **пропозиційними літерами.**

2. Усі літери алфавіту Al є формулами ПМЛ. Якщо A і B є формулами ПМЛ, то $\neg A$, $A \rightarrow B$, $\Box A$ також є формулами ПМЛ.

3. Для довільних формул A, B, C формули ПМЛ:

A1) $A \to (B \to A)$,

A2) $(A \to (B \to C)) \to ((A \to B) \to (A \to C))$,

A3) $(\neg B \to \neg A) \to ((\neg B \to A) \to B)$,

K) $\Box(A \to B) \to (\Box A \to \Box B)$,

є схемами аксіом.

4. Правилами виведення є:

modus ponens (MP): з A і $A \to B$ випливає B,

правило модальної необхідності (RN): з A випливає $\Box A$.

Решта логічних зв'язок вводяться за допомогою вже відомих тотожностей:

$$A \wedge B \leftrightarrow \neg(A \to \neg B), \quad A \vee B \leftrightarrow \neg A \to B,$$
$$A \leftrightarrow B \leftrightarrow (A \to B) \wedge (B \to A).$$

Зауважимо, що як і у випадку ЧВ, нескінченна множина аксіом ПМЛ задається за допомогою чотирьох **схем аксіом** A1), A2), A3), K), кожна з яких породжує нескінченну множину аксіом. Скінченність числа схем аксіом дає можливість для довільної формули легко перевірити, є чи ні вона аксіомою, і, отже, ПМЛ є аксіоматичною теорією.

Визначення 154. **Доведенням у ПМЛ** *називається довільна послідовність формул A_1, A_2,..., A_n цієї теорії така, що кожна формула A_i є або аксіомою, або безпосереднім наслідком попередніх формул цієї послідовності за одним із правил виведення.*
Формула A *називається* теоремою ПМЛ, *якщо існує доведення в* ПМЛ *для A таке, що останнім елементом у цьому доведенні є формула A. Таке доведення називається доведенням формули A в теорії* ПМЛ.

Аксіоматична система ПМЛ, яка включає аксіому K, називається **нормальною модальною логікою.**

Властивості бінарних відношень і різновиди ПМЛ. Якщо задана структура $F = (W, R)$, то залежно від властивостей відношення R на множині W можна отримувати різні ПМЛ. Наприклад:

1) $\forall s \in W\ (sRs)$ — рефлексивність R;
2) $\forall s, t \in W\ (sRt \to tRs)$ — симетричність R;
3) $\forall s \in W\ \exists t \in W\ (sRt)$ — репродуктивність R;
4) $\forall s, t, u \in W\ (sRt \wedge tRu \to sRu)$ — транзитивність R;
5) $\forall s, t, u \in W\ (sRt \wedge sRu \to tRu)$ — евклідовість R;
6) $\forall s, t, u \in W\ (sRt \wedge sRu \to t = u)$ — функціональність часткова R;

7) $\forall s \in W\, \exists! t \in W\, (sRt)$ — функціональність R;
8) $\forall s, t \in W\, (sRt \to \exists u\, (sRu$ і $uRt)$ — слабка щільність;
9) $\forall s, t, u \in W\, (sRt \wedge sRu \to tRu \vee t = u \vee uRt)$ — слабка зв'язність;
10) $\forall s, t, u \in W\, (sRt \wedge sRu \to \exists v(tRv \wedge uRv)$ — слабка скерованість.

Запис $\exists! t$ означає, що «існує один і тільки один елемент t».

Цьому списку властивостей бінарних відношень відповідає такий список схем аксіом ПМЛ:

1) $\Box A \to A$ (T);
2) $A \to \Box\Diamond A$ (B);
3) $\Box A \to \Diamond A$ (D);
4) $\Box A \to \Box\Box A$ (4);
5) $\Diamond A \to \Box\Diamond A$ (5);
6) $\Diamond A \to \Box A$;
7) $\Diamond A \leftrightarrow \Box A$;
8) $\Box\Box A \to \Box A$;
9) $\Box((A \wedge \Box A) \to B) \vee \Box((B \wedge \Box B) \to A)$ (L);
10) $\Diamond\Box A \to \Diamond\Box A$

З правого боку в останньому списку виписані історичні назви відповідних формул. Деякі з цих формул приймаються як схеми аксіом у відповідних аксіоматичних системах. Відповідність між властивостями 1)—10) бінарного відношення R і схемами аксіом 1)—10) випливає з такого твердження.

Теорема 163. *Якщо задана структура $F = (W, R)$, то відношення R має одну із властивостей* 1)—10) *тоді і тільки тоді, коли відповідна схема аксіом є істинною в структурі F* [47].

З цього твердження стає зрозумілим, чому описана вище семантика Кріпке для модальної логіки є такою популярною. З одного боку, ця семантика добре пристосована до різних застосувань, а з другого — важливі властивості відношення R, яке входить до складу означення моделі, підтримується відповідними модальними аксіоматичними системами. Варто зауважити, що існують такі властивості бінарного відношення R, для яких не існує відповідника у вигляді схем модальних формул. Наприклад,

a) $\forall s \in W \neg(sRs)$ — іррефлексивність;
b) $\forall s, t \in W(sRt \wedge tRs \to s = t)$ — антисиметричність;
c) $\forall s, t \in W(sRt \to \neg(tRs))$ — асиметричність.

Існує й така нотація. Якщо $\{\lambda_i | i \in I\}$ довільна множина нормальних логік, то перетин цих логік $\cap\{\lambda_i | i \in I\}$ теж є нормальною логікою. Зокрема, логіка K, визначена за допомогою перетину всіх нор-

мальних логік $K = \cap\lambda_i|\lambda_i$, є мінімальною нормальною модальною логікою. Найменша нормальна логіка, яка включає схеми аксіом $\Sigma_1,\ldots, \Sigma_n$, позначається через $\Lambda = K\Sigma_1 \ldots \Sigma_n$. Беручи до уваги таку нотацію, для деяких відомих ПМЛ прийнято такі позначення:

$$S4 : KT4 \text{ і } S5 : KT45,$$

де K — найменша нормальна логіка, а T, 4 і 5 означають відповідно схеми формул (Т), (4) і (5), які були наведені вище.

Основна властивість мінімальної логіки дає таке твердження:

Теорема 164. *Формула A є теоремою логіки K тоді і тільки тоді, коли A є модальною тавтологією, тобто є істинною в усіх структурах.*

Наступне твердження дає деяку множину теорем нормальної логіки.

Теорема 165. *Для довільних формул A, B, C наведені нижче формули є істинними в кожній нормальній логіці:*

a) $A \to B \vdash \Box A \to \Box B$;
b) $A \to B \vdash \Diamond A \to \Diamond B$;
c) $A \leftrightarrow B \vdash \Box A \leftrightarrow \Box B$;
d) $A \leftrightarrow B \vdash \Diamond A \leftrightarrow \Diamond B$;
e) $\vdash (\Box A \wedge \Box B) \leftrightarrow \Box(A \wedge B)$;
f) $\vdash \Diamond(A \vee B) \leftrightarrow (\Diamond A \vee \Diamond B)$;
g) $\vdash (\Box A \vee \Box B) \to \Box(A \vee B)$;
h) $\vdash \Diamond(A \wedge B) \to (\Diamond A \wedge \Diamond B)$.

Побудуємо доведення тільки для *a*) і *b*), а доведення решти тверджень пропонуються як вправи.

a) $A \to B \vdash \Box A \to \Box B$

1) $A \to B$ — гіпотеза,
2) $\Box(A \to B)$ — RN з 1),
3) $\Box(A \to B) \to (\Box A \to \Box B)$ — аксіома K),
4) $\Box A \to \Box B$ — MP з 2) і 3).

b) $A \to B \vdash \Diamond A \to \Diamond B$

1) $A \to B$ — гіпотеза,
2) $\neg B \to \neg A$ — правило контрапозиції з 1),
3) $\Box(\neg B \to \neg A)$ — RN з 2),
4) $\Box(\neg B \to \neg A) \to (\Box\neg B \to \Box\neg A)$ — аксіома K),
5) $\Box\neg B \to \Box\neg A$ — MP з 3) і 4),
6) $\neg\Box\neg A \to \neg\Box\neg B$ — правило контрапозиції,
7) $\Diamond A \to \Diamond B$ — означення оператора $\Diamond$.■

7.2. Лінійна пропозиційна темпоральна логіка (ЛПТЛ)

У нормальній логіці на відношення R не накладалося жодних обмежень. У застосуваннях часто відношення R є відношенням часткового або лінійного порядку. Якщо відношення R є відношенням лінійного порядку, то отримуємо логіку, яка називається лінійною пропозиційною темпоральною логікою. Відношення лінійного порядку R має таку властивість: для кожного елемента $w \in W$ існує єдиний елемент $w' \in W$ такий, що $(w, w') \in R$. Ця властивість дає можливість ввести ще один модальний оператор, який позначається як $\bigcirc$ і називається X-оператором або **next**-оператором. Неформально $\bigcirc A$ означає, що формула A істинна в наступній точці моделі, яка безпосередньо слідує за даною точкою згідно з відношенням R.

Розглянемо синтаксис і семантику цієї логіки, а також аксіоматичну систему для неї.

Синтаксис. Нехай $Al = \{A, B, C_1,...\}$ — алфавіт атомарних формул, а P — множина пропозиційних формул над цим алфавітом.

Визначення 155 (*синтаксис ЛПТЛ*)
1. *Усі атомарні формули є формулами ЛПТЛ;*
2. *Якщо A і B є формулами ЛПТЛ, то $\neg A$, $A \to B$, $\Box A$, $\bigcirc A$ також є формулами ЛПТЛ.*

Решта логічних зв'язок вводяться за допомогою відомих тотожностей:

$$A \vee B = \neg A \to B,\ A \wedge B = \neg(A \to \neg B),$$
$$A \leftrightarrow B = (A \to B) \wedge (B \to A),\ \Diamond A = \neg\Box\neg A.$$

Семантика. Модель Кріпке для *ЛПТЛ* має вигляд $M = (T, R, f, t_0)$, де множину точок позначають T (а не W) і називають множиною моментів часу, $t_0 \in T$ — початковий момент часу, $R \subseteq T \times T$ бінарне відношення лінійного порядку на T, а для кожного $p \in P$, $f(p)$ є підмножиною множини T тих моментів часу, в які формула p істинна.

Семантика формул *ЛПТЛ* визначається на моделі M таким чином:

1. $M \models p$ тоді і тільки тоді, коли $t_0 \in f(p)$, де $p \in P$;
2. $M \not\models 0$;
3. $M \models (\varphi \to \psi)$ тоді і тільки тоді, коли з $M \models \varphi$ випливає $M \models \psi$;
4. $M \models \Box\varphi$ тоді і тільки тоді, коли $\forall t_1 \in T$ такого, що $(t_0, t_1) \in R$ має місце $t_1 \models \varphi$;
5. $M \models \bigcirc\varphi$ тоді і тільки тоді, коли $\exists t_1 \in T$ таке, що $(t_0, t_1) \in R$ (тобто t_1 безпосередньо слідує за t_0 і $t_1 \models \varphi$).

Аксіоматична система для ЛПТЛ. Для довільних формул A, B, C наступні формули

A1) $A \to (B \to A)$,
A2) $(A \to (B \to C)) \to ((A \to B) \to (A \to C))$,
A3) $(\neg B \to \neg A) \to ((\neg B \to A) \to B)$,
K) $\Box(A \to B) \to (\Box A \to \Box B)$,
T1) $\Diamond A \leftrightarrow \neg\Box\neg A$,
T2) $\bigcirc(A \to B) \to (\bigcirc A \to \bigcirc B)$,
T3) $\Box A \to (A \wedge \bigcirc A \wedge \bigcirc\Box A)$,
T4) $\Box(A \to \bigcirc A) \to (A \to \Box A)$,
T5) $\bigcirc A \leftrightarrow \neg\bigcirc\neg A$

є схемами аксіом.

Правилами виведення є:

modus ponens (MP): з A і $A \to B$ випливає B,

правило модальної необхідності (RN): з A випливає $\Box A$.

Аксіома T4) у наведеній вище системі є аксіомою математичної індукції. Кроком індуктивного доведення є $A \to \bigcirc A$, тобто припускаємо, що A істинна «сьогодні» і доводимо, що A буде істинною «завтра». Якщо індуктивний крок завжди має місце $\Box(A \to \bigcirc A)$, то звідси випливає, що $A \to \Box A$.

Дана аксіоматична система є повною і несуперечною. Зв'язок між синтаксисом і семантикою тверджень *ЛПТЛ* випливає з такої теореми.

Теорема 166. *Формула A є теоремою ЛПТЛ тоді і тільки тоді, коли A є тавтологією ЛПТЛ* [45].

Розглянемо деякі властивості *ЛПТЛ*.

Теорема 167. *Для довільної ЛПТЛ-формули A мають місце такі твердження*:

(a) $\Box A \to A$, (d) $\bigcirc A \to \Diamond A$,
(b) $\Box A \to \bigcirc A$, (e) $\Box A \to \Diamond A$,
(c) $A \to \Diamond A$, (f) $\neg\bigcirc A \leftrightarrow \bigcirc\neg A$.

Доведення.

(a) $\Box A \to A$

1) $\Box A \to (A \wedge \bigcirc A \wedge \bigcirc\Box A)$ — аксіома T3),
2) $(A \wedge \bigcirc A \wedge \bigcirc\Box A) \to A$ — ПК2,
3) $\Box A \to A$ — TI.

(b) $\Box A \to \bigcirc A$

Доведення аналогічне доведенню теореми 167 (а),

(*c*) $A \to \Diamond A$

1) $\Box\neg A \to \neg A$ — теорема 167 (b),
2) $\neg(\neg A) \to \neg\Box\neg A$ — правило контрапозиції,
3) $A \to \Diamond A$ — T1) і ЗПЗ.

(*d*) $\bigcirc A \to \Diamond A$

1) $\Box A \to \bigcirc\neg A$ — теорема 167 (b),
2) $\neg\bigcirc\neg A \to \neg\Box\neg A$ — правило контрапозиції,
3) $\bigcirc A \to \Diamond A$ — аксіоми T5) і T1).

(*e*) $\Box A \to \Diamond A$

1) $\Box A \to A$ — теорема 167 (a),
2) $A \to \Diamond A$ — теорема 167 (c),
3) $\Box A \to \Diamond A$ — TI.

(*f*) $\neg\bigcirc A \leftrightarrow \bigcirc\neg A$

1) $\bigcirc\neg A \leftrightarrow \neg\bigcirc\neg\neg A$ — аксіома T5),
2) $\bigcirc\neg A \leftrightarrow \neg\bigcirc A$ — ЗПЗ. ■

Доведемо ще деякі корисні твердження для цієї логіки.

Теорема 168. *Для довільних ЛПТЛ-формул A і B мають місце такі твердження*:

a) $(A \to \bigcirc A) \vdash (A \to \Box A)$,
b) $(A \to B) \vdash (\Box A \to \Box B)$,
c) $(A \to B) \vdash (\bigcirc A \to \bigcirc B)$.

Доведення.

a) $(A \to \bigcirc A) \vdash (A \to \Box A)$

1) $A \to \bigcirc A$ — гіпотеза,
2) $\Box(A \to \bigcirc A)$ — правило RN з 1),
3) $\Box(A \to \bigcirc A) \to (A \to \Box A)$ — аксіома T4),
4) $A \to \Box A$ — правило MP з 2) і 3).

b) $(A \to B) \vdash (\Box A \to \Box B)$

1) $A \to B$ — гіпотеза,
2) $\Box(A \to B)$ — правило RN з 1),
3) $\Box(A \to B) \to (\Box A \to \Box B)$ — аксіома K),
4) $\Box A \to \Box B$ — правило MP з 2) і 3).

c) $(A \to B) \vdash (\bigcirc A \to \bigcirc B)$

1) $A \to B$ — гіпотеза,

2) $\Box(A \to B)$ — правило RN з 1),
3) $\Box(A \to B) \to O(A \to B)$ — теорема 167 b),
4) $O(A \to B)$ — правило MP з 2) і 3),
5) $O(A \to B) \to (OA \to OB)$ — аксіома T2),
6) $OA \to OB$ — правило MP з 4) і 5).■

Основні властивості *ЛПТЛ* дає наступне твердження.

Теорема 169. *ЛПТЛ є повною і несуперечною логікою.*

Доведення. Несуперечність аксіом $A1$)—$A3$) і правила *MP* випливає з відповідної теореми для числення висловлювань. Покажемо, що аксіоми $T1$)—$T5$) є модальними тавтологіями, оскільки в ПМЛ було показано, що аксіома K) є модальною тавтологією.

a) Аксіома $T1$. Нехай $M = (T, R, f, t_0)$ є довільною структурою і $s \in T$ — довільна точка. Нехай $s \models \Diamond A$ і припустимо, що $s \models \Box\neg A$. Тоді існує точка $s' \in T$ така, що $(s, s') \in R$ і $s' \models A$. Оскільки $s \models \Box\neg A$ для всіх точок, досяжних з точки s за допомогою відношення R, то, зокрема, $s' \models \neg A$. Це означає, що припущення про істинність формули $\Box\neg A$ в точці s є хибним, тобто $s \not\models \Box\neg A$, а це означає, що $s \models \neg\Box\neg A$. Оскільки структура M і точка s довільні, то $\Diamond A \to \neg\Box\neg A$.

Навпаки, нехай $s \models \neg\Box\neg A$. Тоді для всіх точок s', які досяжні з точки s, маємо $s' \not\models \Box\neg A$. Це означає, що для деякого $s' \models A$, але тоді $s \models \Diamond A$, тобто $\neg\Box\neg A \to \Diamond A$.

b) Аксіома $T2$. Нехай $M = (T, R, f, t_0)$ є довільною структурою і $s \in T$ — довільна її точка. Припустимо, що

$$s \models O(A \to B) \text{ і } s \not\models OA \to OB.$$

Тоді існує точка $s' \in T$, безпосередньо досяжна з точки s (тобто $(s, s') \in R$) така, що

$$s' \models A \to B, \quad s' \models A \text{ і } s' \not\models B.$$

Але якщо $A \to B$ і A є істинними формулами в точці s', а B є хибною в цій точці, то $A \to B$ теж має бути хибною формулою і отримуємо суперечність з тим, що $s' \models A \to B$ а це означає, що формула $O(A \to B) \to (OA \to OB)$ є модальною тавтологією. Доведення того, що аксіоми T3)—T5) є модальними тавтологіями, пропонується як вправи.

c) Правило *RN*. Нехай $M = (T, R, f, t_0)$ є довільною структурою, $s \in T$ — довільна її точка і $s \models A$. Необхідно показати, що $s \models \Box A$.

Формула $\Box A$ істинна, якщо для всіх точок s' таких, що $(s, s') \in R^*$ (R^* — рефлексивно-транзитивне замикання відношення R) має місце $s' \models A$. Істинність формули A в кожній такій точці є припущенням,

яке означає, що A є істинною в кожній точці з множини T, а також і в точці s'.

Тепер доведення несуперечності ЛПТЛ випливає з твердження математичної індукції за довжиною доведення в ЛПТЛ. ■

Чергове твердження подає властивості модальних операторів *ЛПТЛ*.

Теорема 170. *Для довільних ЛПТЛ-формул A і B мають місце такі твердження*:

a) $\Box\bigcirc A \leftrightarrow \bigcirc\Box A$,
b) $\Box A \leftrightarrow (A \wedge \bigcirc\Box A)$,
c) $\Diamond A \leftrightarrow (A \vee \bigcirc\Diamond A)$,
d) $\Diamond\bigcirc A \leftrightarrow \bigcirc\Diamond A$,
e) $\bigcirc(A \vee B) \leftrightarrow (\bigcirc A \vee \bigcirc B)$,
f) $\bigcirc(A \wedge B) \leftrightarrow (\bigcirc A \wedge \bigcirc B)$,
g) $\Box(A \wedge B) \leftrightarrow (\Box A \wedge \Box B)$,
h) $(\Box A \vee \Box B) \rightarrow \Box(A \vee B)$,
i) $\Box\Box A \leftrightarrow \Box A$,
j) $\Diamond(A \vee B) \leftrightarrow (\Diamond A \vee \Diamond B)$,
k) $\Diamond(A \wedge B) \rightarrow (\Diamond A \wedge \Diamond B)$,
l) $\Box(A \rightarrow B) \rightarrow (\Diamond A \rightarrow \Diamond B)$,
m) $\Box\Diamond\Box A \leftrightarrow \Diamond\Box A$,
n) $\Diamond\Box\Diamond A \leftrightarrow \Box\Diamond A$,
o) $\Diamond\Box A \rightarrow \Box\Diamond A$.

Доведення.

a) $\Box\bigcirc A \leftrightarrow \bigcirc\Box A$
*a*1) $\Box\bigcirc A \rightarrow \bigcirc\Box A$

1) $\bigcirc A \rightarrow (A \rightarrow \bigcirc A)$ — A1),
2) $\Box(\bigcirc A \rightarrow (A \rightarrow \bigcirc A))$ — RN з 1),
3) $\Box(\bigcirc A \rightarrow (A \rightarrow \bigcirc A)) \rightarrow (\Box\bigcirc A \rightarrow \Box(A \rightarrow \bigcirc A))$ — T4),
4) $\Box\bigcirc A \rightarrow \Box(A \rightarrow \bigcirc A)$ — MP з 2) і 3),
5) $\Box(A \rightarrow \bigcirc A) \rightarrow \bigcirc\Box(A \rightarrow \bigcirc A)$ — T3), ПК2, TI,
6) $\Box\bigcirc A \rightarrow \bigcirc\Box(A \rightarrow \bigcirc A)$ — TI з 5), 6),
7) $\Box\bigcirc A \rightarrow \bigcirc A$ — T3), ПК2, TI,
8) $\Box(A \rightarrow \bigcirc A) \rightarrow (A \rightarrow \Box A)$ — T4),
9) $\Box(\Box(A \rightarrow \bigcirc A) \rightarrow (A \rightarrow \Box A))$ — RN з 8),
10) $\Box(\Box(A \rightarrow \bigcirc A) \rightarrow \bigcirc(\Box(A \rightarrow \bigcirc A) \rightarrow (A \rightarrow \Box A))$ — T3),
11) $\bigcirc(\Box(A \rightarrow \bigcirc A) \rightarrow (A \rightarrow \Box A))$ — MP з 9) і 10),
12) $\bigcirc(\Box(A \rightarrow \bigcirc A) \rightarrow (A \rightarrow \Box A)) \rightarrow$
$\rightarrow (\bigcirc\Box(A \rightarrow \bigcirc A) \rightarrow \bigcirc(A \rightarrow \Box A))$ — T2),
13) $\bigcirc\Box(A \rightarrow \bigcirc A) \rightarrow \bigcirc(A \rightarrow \Box A)$ — MP з 11) і 12),
14) $\bigcirc(A \rightarrow \Box A) \rightarrow (\bigcirc A \rightarrow \bigcirc\Box A)$ — T2),
15) $\bigcirc\Box(A \rightarrow \bigcirc A) \rightarrow (\bigcirc A \rightarrow \bigcirc\Box A)$ — TI з 13) і 14),
16) $\Box\bigcirc A \rightarrow (\bigcirc A \rightarrow \bigcirc\Box A)$ — TI з 6) і 15),
17) $(\Box\bigcirc A \rightarrow (\bigcirc A \rightarrow \bigcirc\Box A)) \rightarrow$
$\rightarrow ((\Box\bigcirc A \rightarrow \bigcirc A) \rightarrow (\Box\bigcirc A \rightarrow \bigcirc\Box A))$ — A2),
18) $(\Box\bigcirc A \rightarrow \bigcirc A) \rightarrow (\Box\bigcirc A \rightarrow \bigcirc\Box A)$ — MP з 16) і 17),
19) $\Box\bigcirc A \rightarrow \bigcirc\Box A$ — MP з 7) і 18).

*a*2) $O\Box A \to \Box OA$

1)	$\Box A \to A$	— теорема 167 (a),
2)	$\Box(\Box A \to A)$	— RN з 1),
3)	$\Box(\Box A \to A) \to O(\Box A \to A)$	— T3), ПК2, TI,
4)	$O(\Box A \to A)$	— MP з 2) і 3),
5)	$O(\Box A \to A) \to (O\Box A \to OA)$	— T2),
6)	$O\Box A \to OA$	— MP з 4) і 5),
7)	$\Box(O\Box A \to OA)$	— RN із 6),
8)	$\Box(O\Box A \to OA) \to (\Box O\Box A \to \Box OA)$	— K),
9)	$\Box O\Box A \to \Box OA$	— MP із 7) і 8),
10)	$\Box A \to O\Box A$	— T3), ПК2, TI,
11)	$\Box(\Box A \to O\Box A)$	— RN з 10),
12)	$\Box(\Box A \to O\Box A) \to O(\Box A \to O\Box A)$	— T3), ПК2, TI,
13)	$O(\Box \to O\Box A)$	— MP з 11) і 12),
14)	$O(\Box A \to O\Box A) \to (O\Box \mathrm{A} \to OO\Box A)$	— T2),
15)	$O\Box A \to OO\Box A$	— MP з 13) і 14),
16)	$\Box(O\Box A \to OO\Box A)$	— RN з 15),
17)	$\Box(O\Box A \to OO\Box A) \to (O\Box A \to \Box O\Box A)$	— T4),
18)	$\Box OA \to \Box O\Box A$	— MP з 16) і 17),
19)	$O\Box A \to \Box OA$	— TI з 18) і 9).

b) $\Box A \leftrightarrow (A \wedge O\Box A)$

*b*1) $\Box A \to (A \wedge O\Box A)$

1)	$\Box A \to (A \wedge OA \wedge O\Box A)$	— аксіома T3),
2)	$(A \wedge OA \wedge O\Box A) \to (A \wedge O\Box A)$	— ПК2,
3)	$\Box A \to (A \wedge O\Box A)$	— TI з 1), 2),

*b*2) $(A \wedge O\Box A) \to \Box A$

1)	$\Box A \to (A \wedge O\Box A)$	— теорема b1),
2)	$O\Box A \to O(A \wedge O\Box A)$	— теорема 168 c)
3)	$(A \wedge O\Box A) \to O(A \wedge O\Box A)$	— теорема 170 b)
4)	$\Box(A \wedge O\Box A) \to O(A \wedge O\Box A)$	— RN з 3),
5)	$\Box(A \wedge O\Box A) \to O(A \wedge O\Box A) \to$ $\to ((A \wedge O\Box A) \to \Box(A \wedge O\Box A))$	— аксіома T4),
6)	$(A \wedge O\Box A) \to \Box(A \wedge O\Box A)$	— MP з 4) і 5),
7)	$(A \wedge O\Box A) \to A$	— ПК2,
8)	$\Box(A \wedge O\Box A) \to A$	— RN із 7),
9)	$(\Box(A \wedge O\Box A) \to A) \to (\Box(A \wedge O\Box A) \to$ $\to \Box A)$	— аксіома *K*),
10)	$\Box(A \wedge O\Box A) \to \Box A$	— MP з 8) і 9),
11)	$(A \wedge O\Box A) \to \Box A$	— TI із 6) і 10).

c) $\Diamond A \leftrightarrow (A \vee O\Diamond A)$

1) $\Box\neg A \leftrightarrow (\neg A \wedge O\Box\neg A)$ — попередня теорема,
2) $\neg(\neg A \wedge O\Box\neg A) \leftrightarrow \neg\Box\neg A$ — пр. контрап.,
3) $A \vee O\neg\Box\neg A) \leftrightarrow A$ — закони де Моргана, аксіоми TI) і T5),
4) $A \vee O\Diamond A) \leftrightarrow \Diamond A$ — аксіома T1).

d) $\Diamond OA \leftrightarrow O\Diamond A$

1) $\Box O\neg A \leftrightarrow O\Box\neg A$ — попередня теорема,
2) $\neg O\Box\neg A \leftrightarrow \neg\Box O\neg A$ — пр. контрап.,
3) $O\neg\Box\neg A \leftrightarrow \neg\Box\neg OA$ — теорема 167 (f),
4) $O\Diamond A \leftrightarrow \Diamond OA$ — аксіома T1).

e) $O(A \vee B) \leftrightarrow (OA \vee OB)$

*e*l) $O(A \vee B) \rightarrow (OA \vee OB)$

1) $O(\neg A \rightarrow B) \rightarrow (O\neg A \rightarrow OB)$ — аксіома T2),
2) $O(\neg A \rightarrow B) \rightarrow (\neg OA \rightarrow OB)$ — теорема 167 (f),
3) $O(A \vee B) \rightarrow (OA \vee OB)$ — тотожність.

*e*2) $(OA \vee OB) \rightarrow O(A \vee B)$

1) $O(\neg A \wedge \neg B) \rightarrow (O\neg A \wedge O\neg B)$ — теорема 167 (f),
2) $\neg(O\neg A \wedge O\neg B) \rightarrow \neg O(\neg A \wedge \neg B)$ — прав. контрап.,
3) $(\neg O\neg A \vee \neg O\neg B) \rightarrow O\neg(\neg A \wedge \neg B)$ — закони де Моргана,
4) $(O\neg\neg A \vee O\neg\neg B) \rightarrow (\neg\neg A \vee \neg\neg B)$ — закони де Моргана, теорема 167 (f),
5) $(OA \vee OB) \rightarrow O(A \vee B)$ — тотожність, ЗПЗ.

f) $O(A \wedge B) \leftrightarrow (OA \wedge OB)$

*f*1) $O(A \wedge B) \rightarrow (OA \wedge OB)$

1) $(A \wedge B) \rightarrow A$ — ПК2,
2) $\Box((A \wedge B) \rightarrow A)$ — RN з 1),
3) $\Box((A \wedge B) \rightarrow A \rightarrow O((A \wedge B) \rightarrow A)$ — теорема 167 (b),
4) $O((A \wedge B) \rightarrow A)$ — MP з 2) і 3),
5) $O((A \wedge B) \rightarrow A) \rightarrow (O(A \wedge B) \rightarrow OA)$ — аксіома T2),
6) $O((A \wedge B) \rightarrow OA)$ — MP з 4) і 5),
7) $O((A \wedge B) \rightarrow OB)$ — аналогічно до 6),
8) $O(A \wedge B) \rightarrow (OA \wedge OB)$ — ПК1 із 6), 7).

*f*2) $(OA \wedge OB) \rightarrow O(A \wedge B)$

1) $O(A \rightarrow \neg B) \rightarrow (OA \rightarrow O\neg B)$ — аксіома T2),
2) $\neg(OA \rightarrow O\neg B) \rightarrow \neg O(A \rightarrow \neg B)$ — пр. контрап.,

3) $\neg(\bigcirc A \to \bigcirc\neg B) \to \bigcirc\neg(A \to \neg B)$ — теорема 167 (f),
4) $(\bigcirc A \wedge \bigcirc B) \to \bigcirc(A \wedge B)$ — еквівалентність.

$g) \Box(A \wedge B) \leftrightarrow (\Box A \wedge \Box B)$

$g1) \Box(A \wedge B) \to (\Box A \wedge \Box B)$

1) $(A \wedge B) \to A$ — ПК2,
2) $\Box((A \wedge B) \to A)$ — RN з 1),
3) $(\Box(A \wedge B) \to A) \to (\Box(A \wedge B) \to \Box A)$ — аксіома K),
4) $(\Box(A \wedge B) \to \Box A)$ — MP з 2) і 3),
5) $(\Box(A \wedge B) \to \Box B)$ — аналогічно до 4),
6) $\Box(A \wedge B) \to (\Box A \wedge \Box B)$ — ПК1 з 4), 5).

$g2) (\Box A \wedge \Box B) \to \Box(A \wedge B)$

1) $(\Box A \wedge \Box B) \to \Box A$ — тавтологія,
2) $\Box A \to \bigcirc A$ — теорема 167 (b),
3) $(\Box A \wedge \Box B) \to \bigcirc A$ — TI з 1) і 2),
4) $(\Box A \wedge \Box B) \to \bigcirc B$ — аналогічно до 3),
5) $(\Box A \wedge \Box B) \to (\bigcirc A \wedge \bigcirc B)$ — ПК1 з 3), 4),
6) $(\bigcirc A \wedge \bigcirc B) \to \bigcirc(A \wedge B)$ — теорема 170 (f),
7) $(\Box A \wedge \Box B) \to \bigcirc(A \wedge B)$ — TI з 5) і 6),
8) $\Box((\Box A \wedge \Box B) \to \bigcirc(A \wedge B))$ — RN із 7),
9) $\Box((\Box A \wedge \Box B) \to \bigcirc(A \wedge B)) \to$ $\to((\Box A \wedge \Box B) \to \Box(A \wedge B))$ — аксіома T4),
10) $(\Box A \wedge \Box B) \to \Box(A \wedge B)$ — MP з 8) і 9).

$h) (\Box A \vee \Box B) \to \Box(A \vee B)$

1) $A \to (A \vee B)$ — ПК1,
2) $\Box(A \to (A \vee B))$ — RN з 1),
3) $\Box(A \to (A \vee B)) \to (\Box A \to \Box(A \vee B))$ — аксіома *K*),
4) $\Box A \to \Box(A \vee B)$ — MP з 2) і 3),
5) $\Box B \to \Box(A \vee B)$ — аналогічно до 4),
6) $(\Box A \vee \Box B) \to \Box(A \vee B)$ — ПВ2 з 4), 5).

$i) \Box\Box A \leftrightarrow \Box A$

$i1) \Box\Box A \to \Box A$

1) $\Box A \to A$ — теорема 167 (a),
2) $\Box(\Box A \to A)$ — RN з 1),
3) $\Box(\Box A \to A) \to (\Box\Box A \to \Box A)$ — аксіома *K*),
4) $\Box\Box A \to \Box A$ — MP з 2) і 3).

*i*2) $\Box A \to \Box\Box A$

1) $\Box A \to (A \wedge \bigcirc A \wedge \bigcirc\Box A)$ — аксіома T3),
2) $(A \wedge \bigcirc A \wedge \bigcirc\Box A) \to \bigcirc\Box A$ — T3), ПК2, TI,
3) $\Box A \to \bigcirc\Box A$ — TI з 1), 2),
4) $\Box(\Box A \to \bigcirc\Box A)$ — RN з 3),
5) $\Box(\Box A \to \bigcirc\Box A) \to (\Box A \to \Box\Box A)$ — аксіома T4),
6) $\Box A \to \Box\Box A$ — MP з 4) і 5).

j) $\Diamond(A \vee B) \leftrightarrow (\Diamond A \vee \Diamond B)$

1) $\Box(\neg A \wedge \neg B) \leftrightarrow (\Box\neg A \wedge \Box\neg B)$ — теорема 170 (g),
2) $\neg(\Box\neg A \wedge \Box\neg B) \leftrightarrow \neg\Box(\neg A \wedge \neg B)$ — пр. контрап.,
3) $(\neg\Box\neg A \vee \neg\Box\neg B) \leftrightarrow \neg\Box\neg(A \vee B)$ — закони де Моргана,
4) $(\Diamond A \vee \Diamond B) \leftrightarrow \Diamond(A \vee B)$ — аксіома TI).

k) $\Diamond(A \wedge B) \to (\Diamond A \wedge \Diamond B)$

1) $(\Box\neg A \vee \Box\neg B) \to \Box(\neg A \vee \neg B)$ — теорема 170 (h),
2) $\neg\Box(\neg A \vee \neg B) \to \neg(\Box\neg A \vee \Box\neg B)$ — прав. контрап.,
3) $(\neg\Box\neg(A \wedge B) \to (\neg\Box\neg A \wedge \neg\Box\neg B)$ — закони де Моргана,
4) $\Diamond(A \wedge B) \to (\Diamond A \wedge \Diamond B)$ — аксіома TI).

l) $\Box(A \to B) \to (\Diamond A \to \Diamond B)$

1) $(A \to B) \to (\neg B \to \neg A)$ — пр. контрап.,
2) $\Box((A \to B) \to (\neg B \to \neg A))$ — RN з 1),
3) $\Box((A \to B) \to (\neg B \to \neg A)) \to$ — аксіома *K*),
$\to (\Box(A \to B) \to \Box(\neg B \to \neg A))$
4) $\Box(A \to B) \to \Box(\neg B \to \neg A)$ — MP з 2) і 3),
5) $\Box(\neg B \to \neg A) \to (\Box\neg B \to \Box\neg A)$ — аксіома *K*),
6) $\Box(A \to B) \to (\Box\neg B \to \Box\neg A)$ — TI з 4), 5),
7) $(\Box\neg B \to \Box\neg A) \to (\neg\Box\neg A \to \neg\Box\neg B)$ — прав. контрап.,
8) $\Box(A \to B) \to (\neg\Box\neg A \to \neg\Box\neg B)$ — TI із 6), 7),
9) $\Box(A \to B) \to (\Diamond A \to \Diamond B)$ — аксіома TI).

m) $\Box\Diamond\Box A \leftrightarrow \Diamond\Box A$

*m*1) $\Box\Diamond\Box A \to \Diamond\Box A$

1) $\Box\Diamond\Box A \to \Diamond\Box A$ — теорема 167 (a).

*m*2) $\Diamond\Box A \to \Box\Diamond\Box A$

1) $\Box A \to \bigcirc\Box A$ — аксіома T3), ПК2, TI,
2) $\Box(\Box A \to \bigcirc\Box A)$ — RN з 1),
3) $\Box(\Box A \to \bigcirc\Box A) \to (\Diamond\Box A \to \Diamond\bigcirc\Box A)$ — теорема 170 (1),
4) $\Diamond\Box A \to \Diamond\bigcirc\Box A$ — MP з 2) і 3),

5) $\Diamond\bigcirc\Box A \to \bigcirc\Diamond\Box A$ — теорема 170 (d),
6) $\Diamond\Box A \to \bigcirc\Diamond\Box A$ — TI з 4) і 5),
7) $\Box(\Diamond\Box A \to \bigcirc\Diamond\Box A)$ — RN із 6),
8) $\Box(\Diamond\Box A \to \bigcirc\Diamond\Box A) \to$ $\to (\Diamond\Box A \to \Box\Diamond\Box A)$ — аксіома T4),
9) $\Diamond\Box A \to \Box\Diamond\Box A$ — MP із 7) і 8).

n) $\Diamond\Box\Diamond A \leftrightarrow \Box\Diamond A$

1) $\Box\Diamond\Box\neg A \leftrightarrow \Diamond\Box\neg A$ — теорема 170 (m),
2) $\Box\neg\neg\Diamond\neg\neg\Box\neg A \leftrightarrow \neg\neg\Diamond\neg\neg\Box\neg A$ — ЗПЗ,
3) $\neg\neg\neg\Diamond\neg\neg\Box\neg A \leftrightarrow \neg\Box\neg\neg\Diamond\neg\neg\Box\neg A$ — прав. контрап. з 2),
4) $\Box\Diamond A \leftrightarrow \Diamond\Box\Diamond A$ — ЗПЗ, аксіома T1).

o) $\Diamond\Box A \to \Box\Diamond A$.

1) $\Box A \to A$ — теорема 167 (a),
2) $\Box(\Box A \to A)$ — RN з 1),
3) $\Box(\Box A \to A) \to (\Diamond\Box A \to \Diamond A)$ — теорема 170 (*l*),
4) $\Diamond\Box A \to \Diamond A$ — MP з 2), 3),
5) $\Box(\Diamond\Box A \to \Diamond A)$ — RN з 4),
6) $\Box(\Diamond\Box A \to \Diamond A) \to (\Box\Diamond\Box A \to \Box\Diamond A)$ — аксіома *K*),
7) $\Box\Diamond\Box A \to \Box\Diamond A$ — MP з 5), 6),
8) $\Diamond\Box A \to \Box\Diamond\Box A$ — теорема 170 (m),
9) $\Diamond\Box A \to \Box\Diamond A$ — TI з 8) і 7). ■

Доведена теорема є основою, на якій ґрунтується метод семантичного табло для ЛПТЛ. Суть цього методу полягає в редукції заданої формули за допомогою еквівалентностей, які були доведені в цій теоремі. Розглянемо в деталях цей метод.

7.3. Метод семантичного табло для ЛПТЛ

Метод семантичного табло використовується в ЛПТЛ, як і в логіці висловлювань, для тестування виконуваності формул шляхом побудови відповідної моделі. Для представлення методу семантичного табло для ЛПТЛ введемо таке означення.

Визначення 156. *Формулу вигляду* $\bigcirc A$ *або* $\neg\bigcirc A$ *називають X-формулою* (*або* **next-time** — *формулою*).

При застосуванні методу семантичного табло формули ЛПТЛ поділяються на такі три класи: α-формули для формул типу кон'юнк-

ції, β-формули для формул типу диз'юнкції і X-формули. Кожному з цих класів відповідають правила розщеплення формули на підформули. Ці правила записуються у вигляді таблиць.

Таблиця правил для α-формул:

α	α_1	α_2
$\neg\neg A$	A	
$A_1 \wedge A_2$	A_1	A_2
$\neg(A_1 \vee A_2)$	$\neg A_1$	$\neg A_2$
$\neg(A_1 \to A_2)$	Az_1	$\neg A_2$
$A_1 \leftrightarrow A_2$	$A_1 \to A_2$	$A_2 \to A_1$
$\Box A$	A	$\bigcirc\Box A$
$\neg\Diamond A$	$\neg A$	$\neg\bigcirc\Diamond A$

Таблиця правил для β-формул:

β	β_1	β_2
$B_1 \vee B_2$	B_1	B_2
$\neg(B_1 \wedge B_2)$	$\neg B_1$	$\neg B_2$
$B_1 \to B_2$	$\neg B_1$	B_2
$\neg(B_1 \leftrightarrow B_2)$	$\neg(B_1 \to B_2)$	$\neg(B_2 \to B_1)$
$\Diamond B$	B	$\bigcirc\Diamond B$
$\neg\Box B$	$\neg B$	$\neg\bigcirc\Box B$

Таблиця правил для X-формул:

X	X_1
$\bigcirc A$	A
$\neg\bigcirc A$	$\neg A$

Семантичне табло. Нехай A є ЛПТЛ-формулою. Кожна вершина семантичного табло T, що відповідає формулі A, позначається деякою множиною підформул формули A. Спочатку T складається з єдиної вершини — початкової вершини, яка позначена одноелементною множиною $\{A\}$, що складається з формули A. Побудова семантичного табло виконується за допомогою такої індуктивної процедури.

Побудова семантичного табло

Беремо деяку непозначену вершину-листок v у графі. Нехай цій вершині відповідає множина формул $U(v)$:

• Якщо $U(v)$ включає пару літер $\{p, \neg p\}$, то

— позначаємо листок v як **замкнутий** '×';

— інакше, якщо такої пари немає і всі формули в $U(v)$ є літерами, то позначаємо листок v як **відкритий** '∘';

• Якщо $U(l)$ не є множиною літер, то беремо формулу з $U(v)$, яка не є літерою або X-формулою.

— Якщо вибрана формула є α-формулою A, то будуємо нову вершину v' як сина вершини v і позначаємо v' множиною

$$U(v') = (U(v) \setminus \{A\}) \cup \{\alpha_1, \alpha_2\}$$

(Якщо A має вигляд $\neg\neg A_1$, то член α_2 відсутній).

— Якщо вибрана формула є β-формулою A, то будуємо дві нові вершини v' і v'' як синів вершини v. Позначаємо v' множиною

$$U(v') = (U(v) \setminus \{A\}) \cup \{\beta_1\},$$

а v'' множиною

$$U(v'') = (U(v) \setminus \{A\}) \cup \{\beta_2\}.$$

• Якщо $U(v)$ складається тільки з літер і X-формул (вершину v у цьому випадку будемо називати X-вершиною), то нехай

$$\{\bigcirc A_1, ..., \bigcirc A_n\}$$

є множиною X-формул в $U(v)$. Будуємо множину формул U

$$U = \{A_1, ..., A_n\}.$$

Якщо U позначає деяку вершину v'', що вже існує, то нічого нового не будуємо, а вершину v'' робимо нащадком вершини v, інакше будуємо нову вершину v' як нащадка v і позначаємо v' множиною формул

$$U(v') = \{A_1, ..., A_n\}.$$

Побудова закінчується тоді, коли всі вершини-листки оброблені й позначені знаками ×, ∘.

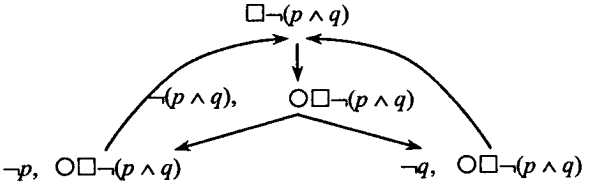
Кінець побудови

Приклад 7.3.2. Розглянемо декілька прикладів побудови семантичного табло.

а) Формула A, яка описує властивість взаємного виключення («mutual exclusion») для формул p і q:

$$A = \Box\neg(p \wedge q).$$

Застосовуючи правила для α-, β- і X-формул, які наведені в таблицях, отримуємо таке семантичне табло для A:

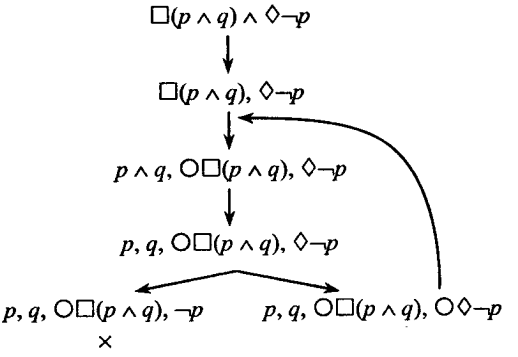


СТ для формули A

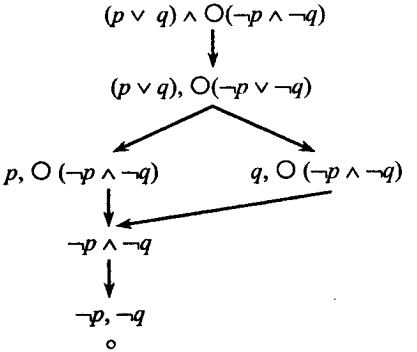
b) Розглянемо ЛПТЛ-формулу, яка після еквівалентних перетворень набуває такого вигляду:

$$\neg(\Box(p \wedge q) \to \Box p) = \Box(p \wedge q) \wedge \neg\Box p = \Box(p \wedge q) \wedge \Diamond\neg p.$$

Семантичне табло для A є таким:



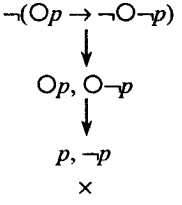
c) Семантичне табло для формули $A = (p \vee q) \wedge \bigcirc(\neg p \wedge \neg q)$ має такий вигляд:



♠

Визначення 157. *Семантичне табло, побудова якого закінчена, називається* **повним**. *Повне семантичне табло називається* **замкнутим**, *якщо воно не має циклів і всі його листки позначені як замкнуті. У протилежному випадку табло називається* **відкритим** (*тобто, коли воно має цикли або деякі його листки позначені як відкриті*).

Приклад 7.3.3. Розглянемо приклад формули, для якої СТ є замкнутим.



СТ є замкнутим. ♠

Теорема 171. *Процедура побудови семантичного табло завжди є скінченною.*

Доведення практично нічим не відрізняється від доведення аналогічного твердження для числення висловлювань. Різниця полягає лише в тому, що підформулами формули, яка відповідає початковій вершині, можуть бути X-формули або формули з модальними операторами. Деталі цього доведення залишаються як вправи. ■

Побудова моделі із семантичного табло. Розглянемо тепер процедуру побудови моделі $M = (W = \{w_1,..., w_n\}, R, f)$ для перевірки виконуваності формули A, виходячи із семантичного табло. Відношення R^*, як і раніше, означає рефлексивно-транзитивне замикання відношення R.

Нехай маємо відкрите семантичне табло T для ЛПТЛ-формули, тоді модель будується за допомогою такої процедури. Слова «шлях у табло» означає шлях через вершини графа, за допомогою якого представлене це табло.

Процедура побудови моделі

• Точками моделі є вершини СТ, позначені як відкриті, і X-вершини, тобто вершини, в яких X-правило було застосоване одночасно до всіх X-формул.

• Для даних двох точок s_1 і s_2 пара $(s_1, s_2) \in R$ тоді і тільки тоді, коли існує шлях із s_1 в s_2 в СТ, що не проходить через інші вершини СТ, які задекларовані як точки моделі.

• $A \in U_i$ тоді і тільки тоді, коли A з'являється в $U(n)$ для деякої вершини СТ n на шляху із точки s_j в точку s_i, який не проходить через інші точки моделі. Формально, нехай $v_1,..., v_m$ — точки моделі, з яких досяжна вершина s_i і

$$\left(v_j = n_{j_1}, n_{j_2}, ..., n_{j_k} = s_i\right), 1 \le j \le m$$

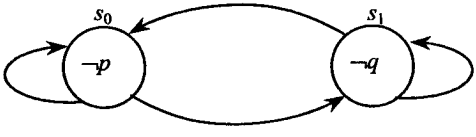
буде деяким шляхом в СТ із вершини v_j у вершину s_i, тоді

$$U_i = \bigcup_{j=1}^{m} U\left(n_{j_2}\right) \cup ... \cup U\left(n_{j_k}\right).$$

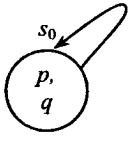
Зокрема, початкова вершина СТ розглядається як перша вершина на шляху при обчисленні U_i.

Кінець побудови

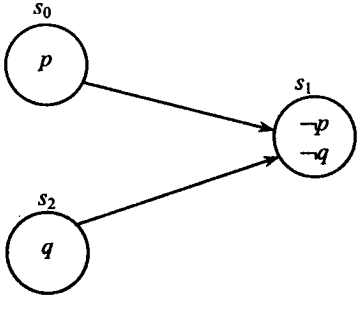
Приклад 7.3.4. a) Модель, яка побудована за СТ для формули a) з прикладу 7.3.2:



b) Модель, побудована за СТ для формули b) з прикладу 7.3.2:



c) Модель, побудована за СТ для формули c) з прикладу 7.3.2:



♠

Визначення 158. *Нехай $M = (W, R, f)$ довільна модель. Модель M називається структурою Хінтікки тоді і тільки тоді, коли для всіх точок $w_i \in W$ і для всіх формул $A \in U_i = f^{-1}(w_i)$ істинно:*
1) якщо A є пропозиційною літерою p, то $\neg p \notin U_i$;
2) якщо A є α-формулою, то $\alpha_1 \in U_i$ і $\alpha_2 \in U_i$;
3) якщо A є β-формулою, то $\beta_1 \in U_i$ або $\beta_2 \in U_i$;
4) якщо $A = \bigcirc A_1$ є X-формулою, то для всіх w_j таких, що $(w_i, w_j) \in R$ $A_1 \in U_j$.

Теорема 172. *Модель, побудована за семантичним табло, є структурою Хінтікки.*

Доведення теореми випливає безпосередньо зі способу побудови СТ і моделі. ■

Визначення 159. *Нехай модель M є структурою Хінтікки. M називається лінійною структурою Хінтікки, якщо відношення R є лінійним, тобто коли для всіх точок w_i існує не більше ніж одна точка w_j така, що $(w_i, w_j) \in R$.*
Нехай M є структурою Хінтікки. M називається виконуючою структурою Хінтікки для формули A, якщо для всіх точок w_i і формули вигляду $A = \Diamond A_1$ маємо:
якщо $A \in U_i$, то для деякого w_j такого, що $(w_i, w_j) \in R^* A_1 \in U_j$.

Теорема 173. *Нехай M є виконуючою лінійною структурою Хінтікки для A. Тоді A виконується.*

Доведення. Розширимо структуру M до інтерпретації за допомогою такого означення для всіх пропозиційних літер p з A:

$$M \models_{w_i} p, \text{ якщо } p \in U_i \text{ або } \neg p \notin U_i,$$

$$M \not\models_{w_i} p \text{ якщо } \neg p \in U_i$$

і доведемо тепер теорему методом математичної індукції за структурою формули A. Доведення зводиться до розгляду окремих випадків. Розглянемо лише один з них, залишаючи решту випадків як вправи.

Нехай $A = \Box A_1 \in U_i$ і нехай w_j довільна точка, така, що $(w_i, w_j) \in R^*$. Тоді існує послідовність точок

$$w_i = w_1, \ldots, w_n = w_j$$

таких, що для довільного k маємо $(w_k, w_{k+1}) \in R$. Для доведення потрібно показати, що $A_1 \in U_k$ для кожного $1 \le k \le n$.

Зауважимо, що для всіх k має місце включення $\Box A_1 \in U_k$. Це легко показати індукцією за числом k. Дійсно, для $k = i$ це вірно за умовою. Використовуючи умову 2) з означення структури Хінтікки і правило для α-формули $\Box A_1$ отримуємо: якщо $\Box A_1 \in U_k$, то $\bigcirc\Box A_1 \in U_k$ і за умовою 4) з означення структури Хінтікки отримуємо $\Box A_1 \in U_{k+1}$.

Тепер, якщо $\Box A_1 \in U_k$, то за умовою 2) з означення структури Хінтікки маємо $A_1 \in U_k$. З припущення індукції за структурою формули випливає $M \models_{w_i} A_1$. А з того, що w_j довільна точка випливає $M \models_{w_i} \Box A_1$. ■

Теорема 174 *Якщо формула A* **ЛПТЛ** *є виконуваною, то виконуюча лінійна структура Хінтікки для A може бути виділена з моделі, що збудована за СТ.*

Доведення. Нехай формулу A в процесі розщеплення зведено до множини формул, яка включає формули $\Diamond A_1, \ldots, \Diamond A_i, \Diamond A_{i+1}, \ldots, \Diamond A_n$. Для доведення теореми потрібно показати, яким чином можна побудувати шлях, виходячи із структури, так щоб кожна формула вигляду $\Diamond A_j$, ($j = 1, 2, \ldots, n$), що асоціюється з цим шляхом, виконувалася. Для цього створимо список L, який поділимо на дві частини. До першої частини віднесемо формули, які ще не виконані, а до другої — ті, що виконані:

$$L = \{\Diamond A_1, \ldots, \Diamond A_i | \Diamond A_{i+1}, \ldots, \Diamond A_n\}.$$

Спочатку шлях складається тільки з однієї точки $l_0 = w_0$ і не має жодної виконаної формули:

$$L_0 = \{\Diamond A_1, \ldots, \Diamond A_n | \}.$$

Припустимо, що такий шлях уже збудований для точки l_m і нехай $\Diamond A_1$ — перша з формул, яка ще не виконана в L_m. Тоді згідно з процедурою побудови СТ має існувати точка w, досяжна по шляху

$$l_m, \ldots, l_{m'} = w$$

така, що виконує $\Diamond A_1$. Розширивши шлях $l_0, \ldots, l_m$ за допомогою конкатенації його зі шляхом $l_{m+1}, \ldots, l_{m'}$, отримуємо шлях

$$l_0, \ldots, l_m, l_{m+1}, \ldots, l_{m'} = w.$$

Множина $L_{m'}$ визначається шляхом переносу формули $\Diamond A_1$ у праву частину списку з його лівої частини, додаючи до списку всі нові формули, що з'явилися на шляху $l_m, \ldots, l_{m'}$ і яких ще немає в списку $L_{m'}$. Формально цей крок доведення є таким.

Поділимо $\{\Diamond A_1, ..., \Diamond A_i \mid \Diamond A_{i+1}, ..., \Diamond A_n\}$ на дві підмножини: $\{\Diamond B_1, ..., \Diamond B_j\}$ — формули, які все ще не виконані, і $\{\Diamond C_1, ..., \Diamond C_k\}$ — нові формули, що з'явилися на цьому шляху і які не виконані. Тоді

$$L_{m'} = \{\Diamond B_1, ..., \Diamond B_j, \Diamond C_1, ..., \Diamond C_k \mid \Diamond A_{i+1}, ..., \Diamond A_n, \Diamond B_1', ..., \Diamond B_j'\}.$$

Оскільки існує лише скінченне число таких формул, що можуть з'явитися в структурі Хінтікки для довільної формули A, і кожен крок принаймні одну з формул виконує, то наступить такий момент, коли ліва частина списку стане пустою. А це означає, що кожна формула вигляду $\Diamond A_j$ виконується на цьому шляху.

Нарешті зазначимо, що кожний шлях у структурі Хінтікки є лінійною структурою. ■

З двох попередніх теорем випливає така.

Теорема 175. *ЛПТЛ-формула виконується тоді і тільки тоді, коли для неї існує виконуюча структура Хінтікки.*

Доведення. Якщо ЛПТЛ-формула виконується, то за теоремою 174 виконуюча лінійна структура Хінтікки для цієї формули існує і будується за її СТ.

Навпаки, якщо виконуюча лінійна структура Хінтікки для ЛПТЛ-формули існує, то за теоремою 173 ця формула виконується. ■

Наслідок 49. *Метод СТ є процедурою виконуваності для ЛПТЛ-формул.*

Доведення. Будуємо СТ для формули A. Якщо СТ замкнуте, то формула A не виконується. Якщо існує хоча б одна відкрита вершина-листок, то формула A виконується. У протилежному випадку СТ має цикли і для виконуваності потрібно знайти лінійну виконуючу структуру Хінтікки для A. Якщо такої структури не існує, то A не виконується. ■

Наслідок 50. *ЛПТЛ-формула є виконуваною тоді і тільки тоді, коли вона виконується на скінченній моделі.*

Доведення очевидним чином випливає з наведених вище тверджень. ■

7.4. Приклад застосування ЛПТЛ

ЛПТЛ є формальною логічною мовою, за допомогою якої можна записати і довести деякі властивості систем, які включають паралельні обчислення. Розглянемо один приклад застосування ЛПТЛ для специфікації паралельного алгоритму.

Проблема взаємного виключення. Проблема взаємного виключення часто з'являється в системах паралельних програм. Припустимо, що декілька паралельних процесів потребують доступу до єдиного спільного ресурсу (наприклад, декілька персональних комп'ютерів, що працюють у мережі, потребують доступу до принтера). Цей доступ завжди можливий, якщо можливе взаємне виключення: у кожний момент часу тільки один з процесів може використовувати спільний ресурс.

Застосування ЛПТЛ для специфікації подібного типу розподілених обчислень полягає в тому, що за допомогою формул цієї логіки можна виразити такі властивості реальних систем як **mutex** (взаємне виключення), **fairness** (якщо якась подія має відбутися, то настане такий момент часу, коли вона відбудеться) і т. д. Наприклад, різниця між формулами

$$\Box\bigcirc p \wedge \Box\bigcirc q \text{ і } \Diamond\Box p \wedge \Box\bigcirc q$$

полягає в тому, що перша формула свідчить про те, що p і q є істинними нескінченно часто і, зокрема, p може бути істинною в ті моменти часу, коли q є хибною, і навпаки. Друга формула свідчить про те, що p колись буде істинною, і в кожний такий момент часу колись буде істинною формула q.

Ця різниця відіграє ключову роль у застосуванні ЛПТЛ при формалізації властивості **fairness**:

• **Слабка fairness**: якщо деякий процесс P буде колись активним нескінченно часто, то завжди можливий початок виконання цього процесу ($\Diamond\Box active(P) \rightarrow \Box\Diamond execution(P)$).

• **Сильна fairness**: якщо деякий процес P буде активним нескінченно часто, то завжди можливий початок виконання цього процесу ($\Box\Diamond active(P) \rightarrow \Box\Diamond execution(P)$).

Інша важлива властивість розподілених програм, яка називається *liveness*, теж виражається за допомогою ЛПТЛ:

• **liveness**: якщо деякий процес розподіленої (паралельної) програми хоче почати працювати, то колись настане такий момент часу, в який цей процес буде працювати ($\Box(want(P) \rightarrow \Diamond work(P))$).

Алгоритм Петерсона взаємного виключення. Алгоритм Петерсона [56] дає можливість розв'язати проблему взаємного виключення в окремому випадку для двох процесів P_1 і P_2, які взаємодіють між собою за допомогою *розділених змінних*. Тут нас не цікавить те, що відбувається всередині процесів, а тільки взаємодія, яка гарантує взаємне виключення. Цей механізм використовує три розділені змінні: дві з них $C1$ і $C2$ є булевими, а третя змінна *last* є змінною, що набуває два значення 1 і 2.

Алгоритм Петерсона

(* глобальні змінні P_1, P_2, *last* *)
var $C1$, $C2$: **0..1** : = **0; last 1..2;**

Процес P_1:
while true do
  begin
    a_1 : некритична-секція 1;
    b_1 : $C1 := 1$;
    c_1 : *last* := 1;
    d_1 : *repeat until* ($C2 = 0 \vee last = 2$)
    e_1 : критична-секція 1;
    f_1 : $C1 := 0$;
  end;
end.

Процес P_2:
while true do
  begin
    a_2 : некритична-секція 2;
    b_2 : $C2 := 1$;
    c_2 : *last* := 2;
    d_2 : *repeat until* ($C1 = 0 \vee last = 1$)
    e_2 : критична-секція 2;
    h_2 : $C2 := 0$
  end;
end.

Алгоритм Петерсона є симетричним: перехід від процесу P_1 до процесу P_2, і назад реалізується заміною змінної $C1$ на $C2$, а також заміною значень 1 на 2 для змінної *last*. Алгоритм працює таким чином. Розглянемо, наприклад, процес P_1. У початковий момент часу P_1 працює і не потребує доступу до спільного ресурсу. У цьому випадку говорять, що процес знаходиться в *некритичній секції*. Якщо спільний ресурс потрібний, то спочатку виконується *протокол входу*: змінна $C1$ набуває значення 1, що свідчить про те, що процес P_1 потребує спільний ресурс. Потім змінна *last* набуває значення 1 і процес P_1 отримує дозвіл на доступ до спільного ресурсу. Цей доступ затримується до тих пір, доки значення змінної *last* не буде дорівнювати 2 або доки $C2$ не стане рівним 0. $C2 = 0$ означає, що процес P_2 останнім використовував спільний ресурс. Після цього процес P_1 входить до своєї критичної секції і використовує спільний ресурс.

Кожний з процесів, який увійшов до критичної секції, має колись вийти з неї. Отже, алгоритм Петерсона має задовольняти таким двом умовам:

- **mutex:** тільки один процес P_1 або P_2 може знаходитися у своїй критичній секції;
- **liveness:** якщо процес P_1 або P_2 хоче ввійти до своєї критичної секції, то він колись ввійде до цієї секції.

Отже, правильність алгоритму Петерсона можна описати за допомогою таких формул:

$$\Box\neg(at(e_1) \wedge at(e_2)) \quad \text{(mutex)}$$
$$\Box((at(b_1) \rightarrow \Diamond at(e_1)) \quad \text{(liveness)}$$
$$\Box((at(b_2) \rightarrow \Diamond at(e_2)) \quad \text{(liveness),}$$

де $at(s)$ означає виконання оператора s.

Для доведення правильності цього алгоритму потрібно ввести аксіоми, які диктуються операторами мови програмування. В алгоритмі Петерсона таких операторів три: присвоювання, умовний та циклу. Для цих операторів введемо такі очевидні аксіоми, які будемо називати **Аксіомами Ходу Обчислень (АХО)** [45]:

а) оператору присвоювання

$$l_i : x := ; l_{i+1} : ...$$

відповідає аксіома

$$\vdash l_i \to \Diamond l_{i+1};$$

b) умовному оператору

$$l_i : if\ B\ then\ (l_t :)\ S1\ else\ (l_f :)\ S2$$

відповідають аксіоми

$$\vdash (l_i \wedge \Box B) \to \Diamond l_t \text{ і } \vdash (l_i \wedge \Box \neg B) \to \Diamond l_f;$$

c) оператору циклу

$$l_i : while\ B\ do\ (l_t :)\ S1\ od\ (l_f :)\ S2$$

відповідають аксіоми

$$\vdash (l_i \wedge \Box B) \to \Diamond l_t \text{ і } \vdash (l_i \wedge \Box \neg B) \to \Diamond l_f.$$

Очевидно, що істинною є формула $l_i \to (l_t \vee l_f)$ для умовного оператора та оператора циклу.

Скористаємося цими аксіомами для доведення правильності алгоритму Петерсона. Доведення правильності опирається на такі дві леми.

Лема 8. (a) $\vdash \Box\ ((last = 1) \vee (last = 2))$;
(b) $\vdash \Box(C1 = 1) \Leftrightarrow (at(d_1) \vee at(e_1) \vee at(f_1))$;
(c) $\vdash \Box(C2 = 1) \Leftrightarrow (at(d_2) \vee at(e_2) \vee at(f_2))$.

Доведення. Доведемо правильність формули (b), а доведення решти формул пропонується виконати як вправи. Істиність формули (b) початково виконується, оскільки $C1$, а також і $at(d_1) \vee at\ (e_1) \vee at(f_1)$ є хибними (у цьому випадку обчислення знаходяться в точці a_1 алгоритму Петерсона). Припустимо, що (b) виконується, тоді необхідно показати, що при виконанні кожного із десяти операторів алгоритму Петерсона ця формула залишається істинною. Якщо, наприклад, буде виконаний оператор b_1, то $C1$ разом з $at(e_1)$ будуть істинними, а тоді і істинність усієї формули матиме місце. Аналогічним чином можна переконатися в істинності цієї формули і для решти операторів. ■

Лема 9. (a) $\vdash \Box(\Box\neg C1 \vee \Diamond(last = 1))$;
(b) $\vdash \Box(\Box\neg C2 \vee \Diamond(last = 2))$.

Доведення. Доведемо справедливість формули (b), а доведення формули (a) є аналогічним, у силу симетричності.

1) $\vdash \Box\ at(a_2) \vee \Diamond at(c_2)$ — дефініція некритичної секції;
2) $\vdash \Box\ at(a_2) \to \Box\neg(at(d_2) \vee last = 2)$ — АХО;
3) $\vdash \Box\neg C2 \vee \Diamond at(d_2)$ — лема 8 (c) з 1), 2);
4) $\vdash at(c_2) \to \Diamond\ (at(d_2) \wedge last = 2)$ — АХО;
5) $\vdash \Box\neg C2 \vee \Diamond(last = 2)$ — теорема 170 k), 1) з 3, 4;
6) $\vdash \Box(\Box\neg C2 \vee \Diamond(last = 2)$ — RN із 5). ■

Якщо покажемо, що $\vdash at(d_1) \to \Diamond at(e_1)$, то після застосування Аксіом Ходу Обчислень і правила модальної необхідності отримаємо доведення властивості *liveness* для процесу P_1. А доведення живості для процесу P_2 виконується аналогічно.

Теорема 176. $\vdash at(d_1) \to \Diamond at(e_1)$.

Доведення.

1) $\vdash \Box at(d_1) \to \Diamond C2$ — АХО;
2) $\vdash \Box at(d_1) \to \Diamond(last = 2)$ — лема 9 (b) з 1);
3) $\vdash \Box at(d_1) \to \Diamond\Box(last = 2)$ — лема 8 (a) і АХО з 2);
4) $\vdash \Box at(d_1) \to \Diamond(last = 1)$ — АХО;
5) $\vdash \Box\Box at(d_1) \to \Box\Diamond(last = 1)$ — RN з 4) і аксіома T1);
6) $\vdash \Box at(d_1) \to \Box\Box at(d_1)$ — тавтологія;
7) $\vdash \Box at(d_1) \to \Box\Diamond(last = 1)$ — TI з 5) і 6);
8) $\vdash \Diamond\Box(last = 2) \to \Box\Diamond(last = 2)$ — тавтологія;
9) $\vdash \Box at(d_1) \to \Box\Diamond(last = 2)$ — TI з 3) і 8);
10) $\vdash \Box at(d_1) \to \Box\Diamond(last = 1 \wedge last = 2)$ — ПК із 7) і 9);
11) $\vdash \Box at(d_1) \to 0$ — з 10) оскільки last = 1 або 2;
12) $\vdash \Box at(d_1) \to \Diamond at(e_1)$ — АХО. ■

7.5. Розширення ЛПТЛ (РЛПТЛ)

Часто в застосуваннях виникає потреба у використанні ЛПТЛ-формули A, яка є істинною на деякому інтервалі $[w, w']$ доти, доки є хибною формула B. В ЛПТЛ записати це за допомогою лише темпоральних операторів $\Box$, $\Diamond$, $\bigcirc$ неможливо. Для того, щоб це можна було зробити, використовують оператори типу «until». Таких операторів є два і їх часто називають «слабким until» і «сильним until» і позначають відповідно літерами W і U.

Визначення 160 (*слабкий until*). $w_0 \models AWB$, *якщо*
- $\forall i \in N, w_i \models A$ *або*
- $\exists n$: $w_n \models B$ *і* $\forall (0 \le i < n)\ w_i \models A$.

Приклад 7.5.5. Формула $\Box(p \to \bigcirc(\neg qWr))$ є істинною, якщо з істинності формули p в деякій точці w безпосередньо випливає, що

a) формула q буде хибна в точці, яка є наступною за точкою w, поки буде істинна формула r або

b) формула q є завжди хибна, починаючи з наступної точки, яка йде за точкою w. ♠

Визначення 161 (*сильний until*). $w_0 \models A\mathbf{U}B$, *якщо*
- $\exists n \in N$: $w_n \models B$ *і* $\forall (0 \le i < n)\ w_i \models A$.

Темпоральні оператори $\Diamond$ і $\Box$ можна визначити за допомогою операторів W і $\mathbf{U}$ таким чином:

$$\Diamond p \leftrightarrow 1\mathbf{U}p$$
$$\Box p \leftrightarrow pW0.$$

Аксіоматична система для РЛПТЛ. Для довільних формул A, B, C схеми формул:

A1) $A \to (B \to A)$,
A2) $(A \to (B \to C)) \to ((A \to B) \to (A \to C))$,
A3) $(\neg B \to \neg A) \to ((\neg B \to A) \to B)$,
K) $\Box(A \to B) \to (\Box A \to \Box B)$,
T1) $\Diamond A \leftrightarrow \neg\Box\neg A$,
T2) $\bigcirc(A \to B) \to (\bigcirc A \to \bigcirc B)$,
T3) $\Box A \to (A \wedge \bigcirc A \wedge \bigcirc\Box A)$,
T4) $\Box(A \to \bigcirc A) \to (A \to \Box A)$,
T5) $\bigcirc A \leftrightarrow \neg\bigcirc\neg A$,
T6) $AUB \leftrightarrow (B \vee (A \wedge \bigcirc(AUB))$
T7) $AUB \to \Diamond B$.

є схемами аксіом.

Правилами виведення є:

modus ponens (MP): з A і $A \to B$ випливає B,
правило модальної необхідності (RN): з A випливає $\Box A$.

Основна властивість цієї аксіоматичної системи випливає з наступної теореми:

Теорема 177. *Формула A є теоремою РЛПТЛ тоді і тільки тоді, коли вона — тавтологія РЛПТЛ.*

Більш глибокі зв'язки введених операторів з темпоральними операторами випливають з такої теореми.

Теорема 178. *Для довільних формул A і B мають місце такі твердження:*

a) $A\mathbf{U}B \leftrightarrow (AWB) \wedge \Diamond B$,
b) $AWB \leftrightarrow (A\mathbf{U}B) \vee \Box A$,
c) $\neg(A\mathbf{U}B) \leftrightarrow (\neg B)W(\neg A \wedge \neg B)$,
d) $\neg(AWB) \leftrightarrow (\neg \mathrm{B})\mathbf{U}(\neg A \wedge \neg B)$.

Доведення цієї теореми пропонується як вправа.

Існують і інші більш виразні модальні логіки, наприклад, *CTL*, *CTL**, μ-числення та ін.

7.6. Контрольні питання, задачі і вправи

1. Дайте означення синтаксису і семантики ПМЛ.
2. Коли ПМЛ називається лінійною?
3. Дайте інтерпретацію аксіом Т1—Т5 для ЛПТЛ словами.
4. Довести такі формули в ЛПТЛ:
   a) $\Box A \leftrightarrow (A \wedge \bigcirc\Box A)$;
   b) $\Diamond A \leftrightarrow (A \vee \bigcirc\Diamond A)$.
5. Довести за допомогою СТ, що:
   a) усі аксіоми Т1)—Т5) є тавтологіями ЛПТЛ;
   b) усі формули з твердження 170 є тавтологіями ЛПТЛ.
6. Побудувати СТ для формули

$$\Box(\Diamond(p \wedge q) \wedge \Diamond(\neg p \wedge q) \wedge \Diamond(p \wedge \neg q)).$$

7. Довести, що для алгоритму Петерсона правдиві такі формули:

$$\Box(at(b_2) \rightarrow at(e_2));$$
$$\Box\neg(at(e_1) \wedge at(e_2)).$$

8. Довести, що формули
   a) $\Box(p \vee q) \rightarrow (\Box p \vee \Box q)$;
   b) $\Diamond p \rightarrow \bigcirc p$;
   c) $\Diamond p \rightarrow p$;
   d) $\Diamond p \rightarrow \Box p$

не є тавтологіями ЛПТЛ.

8 ЧИСЛЕННЯ ПРЕДИКАТІВ ПЕРШОГО ПОРЯДКУ

Існують такі види логічних формул, які не можна записати у вигляді формул числення висловлювань і формул модальної логіки. Наприклад:

(1) Усі риби, крім акул, добре відносяться до дітей;

(2) Усі люди смертні. Сократ — людина. Отже, Сократ смертний;

(3) Кожен, хто дружить з Ярославом, є другом Ігоря. Олег не є другом Ігоря. Отже, Олег не є другом Ярослава.

Коректність таких висновків ґрунтується не тільки на істинності відповідних функціональних відношень, а також і на розумінні таких слів, як «усі», «кожен» і т. д.

Для того, щоб зробити більш зрозумілою структуру складних висловлювань, користуються спеціальною мовою — **мовою предикатів першого порядку (МППП).** Ця мова є розширенням мови числення висловлювань і мови термів, до означення якої і переходимо.

8.1. Синтаксис: алфавіт і формули

Розширимо алфавіт функціональних символів мови термів символами $A_1^{n_1}$, $A_2^{n_2}$,..., $A_k^{n_k}$,... які будуть називатися предикатними символами. Верхній індекс предикатного символа, як і раніше, означає його арність. Множину предикатних символів будемо називати сигнатурою предикатів і позначати її **П**.

Предикатні символи, коли їх застосувати до термів, породжують елементарні (атомарні) формули, або точніше: якщо A^n — предикатний символ, а t_1, t_2,..., t_n — терми, то $A^n(t_1, t_2, \ldots, t_n)$ — **елементарна формула.**

Формули мови предикатів першого порядку визначаються так.

Визначення 162. *(a) Кожна елементарна формула є формулою МППП;*
(b) якщо A і B — формули і x — предметна змінна, то кожний із виразів $\neg A$, $A \to B$, $(\forall x)\, A$ *є формулою МППП;*
(c) вираз є формулою МППП тоді і тільки тоді, коли це випливає з пунктів (a), (b).

Формули $A \wedge B$, $A \vee B$, $A \leftrightarrow B$ визначаються так само, як це робилося в численні висловлювань:

Символи $\neg$, $\to$, $\wedge$, $\vee$, $\leftrightarrow$ називаються логічними зв'язками, а $\forall$ — квантором загальності. Часто ще розглядають квантор існування $(\exists x)A$, який визначається як $\neg(\forall x)\,(\neg A)$. Формула $(\forall x)A$ читається «для всіх x має місце A», а формула $(\exists x)A$ — «існує x такий, що має місце A».

Зауважимо, що логічні зв'язки мають ранги, про які йшлося в численні висловлювань, але, крім того, умовимося, що квантори $\forall y$ і $\exists y$ розміщуються за рангами між зв'язками $\leftrightarrow$, $\to$ і $\vee$, $\wedge$, $\neg$. Умовимося також опускати дужки, в яких знаходиться формула $Q\,A$ у формулах вигляду $Q_1(QA)$, де Q і Q_1 — довільні квантори. Наприклад, замість $(\forall x_1(\exists x_2(\forall x_4\, A_1^3(x_1, x_2, x_4))))$ записується формула $\forall x_1 \exists x_2 \forall x_4\, A_1^3(x_1, x_2, x_4)$.

Введемо поняття вільного і зв'язаного входження змінної у формулу. У виразі $(\forall x)A$ $((\exists x)A)$ формула A називається **областю дії квантора** $\forall x(\exists x)$. Слід зазначити, що формула A може і не мати входжень змінної x. У такому випадку вважається, що формули A і $\forall x A$ однакові.

Визначення 163. *Входження змінної x у дану формулу A називається* **зв'язаним**, *якщо x — змінна квантора* $\forall x$ *або* $\exists x$, *який входить у формулу A або знаходиться в області дії квантора* $\forall x$ *чи* $\exists x$, *який входить у формулу A; у протилежному випадку входження змінної x у дану формулу називається* **вільним**.

Змінна називається **вільною (зв'язаною)** змінною у даній формулі, якщо існують вільні (зв'язані) її входження в цю формулу. Отже, змінна може бути одночасно вільною і зв'язаною в одній і тій самій формулі. Формула називається **замкнутою**, якщо вона не має вільних змінних.

Приклад 8.1.1. Нехай маємо такі формули:

1) $A_1^2(x, y)$,

2) $A_1^2(x, y) \to \forall x\, A_1^1(x)$,

3) $\forall x(A_1^2(x, y) \to \forall x\, A_1^1(x))$.

Єдине входження змінної x у формулу 1) вільне. Перше входження змінної x у формулу 2) вільне, а друге і третє зв'язані. Всі входження змінної x у формулу 3) є зв'язаними. Слід зауважити, що одна і та сама змінна може мати вільні і зв'язані входження в одну і ту саму формулу, як це мало місце у формулі 2). Зазначимо також, що входження змінної може бути зв'язаним в тій чи іншій формулі A і в той же час вільним у деякій підформулі формули A. Наприклад, перше входження x у формулу 2) вільне, але формула 2) є підформулою формули 3), де те саме входження x уже зв'язане. ♠

Визначення 164. *Терм t називається* **вільним** *для змінної x у формулі A, якщо ніяке вільне входження x в A не лежить в області дії ніякого квантора $\forall y$, де y — змінна, що входить до терму t.*

Приклад 8.1.2. 1) Терм x вільний для змінної y в $A^1(y)$, але не є вільним в $\forall x A^1(y)$. Терм $f(x, z)$ вільний для змінної x в $\forall y A^2(x, y) \to B^1(x)$, але не є вільним для x в $\exists z \forall y (A^2(x, y) \to B^1(x))$.

2) Довільний терм, який не має змінних (константний терм), вільний для кожної змінної в довільній формулі.

3) Терм t вільний для довільної змінної у формулі A, якщо ніяка змінна терму t не є зв'язаною змінною у формулі A.

4) Терм x вільний для x у довільній формулі.

5) Довільний терм вільний для x у формулі A, коли A не включає вільних входжень x. ♠

8.2. Семантика: виконуваність, інтерпретації, моделі

Формули мають сенс тільки тоді, коли існує будь-яка інтерпретація символів, що входять до цих формул. Для цього перейдемо до означення виконуваності й істинності формул МППП, які будемо формулювати відповідно до індуктивних кроків означення формул МППП.

Нехай маємо деяку інтерпретацію, тобто систему, яка складається з деякої непустої множини D — області інтерпретації — і відповідності h, що ставить кожному предикатному символу A_j^n деяке n-арне відношення на D, кожному функціональному символу f_j^n — деяку n-арну операцію на D і кожній предметній константі a — деякий елемент із D. При заданій інтерпретації предметні змінні уявляються такими, що пробігають область D цієї інтерпретації.

Приклад 8.2.3. Розглянемо формули

1) $A_1^2(x, y)$,

2) $\forall y\, A_1^2(x, y)$,

3) $\exists y \forall x\, A_1^2(y, x)$.

Якщо за область інтепретації взяти множину натуральних чисел N^+, а формулу $A_1^2(x, y)$ інтерпретувати як $x \le y$, то

1) являє собою відношення $x \le y$, яке виконується на всіх упорядкованих парах чисел (a, b), таких що $a \le b$;

2) являє собою відношення з одним аргументом x вигляду «для кожного натурального числа y, $x \le y$», яке виконується лише для числа 1;

3) буде істинним висловлюванням, яке стверджує існування мінімального цілого натурального числа. Якби областю інтерпретації служила множина всіх цілих чисел, то формула 3) була б хибною. ♠

Нехай задано деяку інтерпретацію з областю D, і нехай Σ — множина всіх зліченних послідовностей елементів із D. Визначимо тепер, що означає, що формула A виконана на послідовності $s = (b_1, b_2, \ldots)$ із Σ при заданій інтерпретації.

Нехай s^* означає функцію одного аргументу, яка визначена на множині термів і набуває своїх значень в області D деякої інтерпретації h, тобто s^*: $T(\Omega, X) \to D$. Ця функція визначається так:

а) Якщо терм t — предметна змінна x_i, то $s^*(t) = b_i$;

б) Якщо терм t — предметна константа, то $s^*(t) = h(t)$, тобто $s^*(t)$ збігається з інтерпретацією цієї константи в області D;

в) Якщо f^n — функціональний символ арності n, якому при інтерпретації h відповідає операція ω^n на області D, і $t_1, t_2, \ldots, t_n$ — терми, то $s^*(f^n(t_1, t_2, \ldots, t_n)) = \omega(s^*(t_1), s^*(t_2), \ldots, s^*(t_n))$.

Отже, якщо говорити неформально, то для довільної послідовності $s = (b_1, b_2, \ldots)$ і довільного терму t $s^*(t)$ є елементом множини D, який отриманий шляхом підстановки при кожному i елемента b_i замість усіх входжень змінної x_i в терм t і виконання після цього всіх операцій інтерпретації h, які відповідають функціональним символам терму t.

Приклад 8.2.4. Нехай $t = f_2^2(x_4, f_1^2(x_2, a))$, $D = Z$ — множина цілих чисел, а f_2^2 і f_1^2 інтерпретуються як операції множення та додавання відповідно, і $a = 1$, то для довільної послідовності цілих чисел $s = (b_1, b_2, b_3, b_4 \ldots)$ $s^*(t)$ являє собою ціле число $b_4 \cdot (b_2 + 1)$. ♠

Визначення 165. *Якщо A є елементарною формулою $A_j^n(t_1, t_2, \ldots, t_n)$ і R — відношення, що відповідає цій формулі на D при заданій інтерпретації, то формула A називається виконаною на послідовності $s = (b_1, b_2, \ldots, b_n, \ldots)$ тоді і тільки тоді, коли $(s^*(t_1), s^*(t_2), \ldots, s^*(t_n)) \in R$.*

Формула $\neg A$ виконана на послідовності s тоді і тільки тоді, коли формула A не виконана на s.

Формула $A \to B$ виконана на s тоді і тільки тоді, коли формула A не виконана на s або коли формула B виконана на s.

Формула $\forall x_i A$ виконана на $s = (b_1, b_2, \ldots)$ тоді і тільки тоді, коли формула A виконується на довільних послідовностях із Σ, які відрізняються одна від одної не більше ніж своїми i-ми компонентами.

Формула A називається виконуваною (або такою, що виконується), якщо існує інтерпретація h, при якій вона виконується хоча б на одній послідовності інтерпретації h.

Формула A називається **істинною** *при заданій інтерпретації h, якщо вона виконується на кожній послідовності із Σ, і* **хибною**, *якщо вона не виконується на жодній послідовності із Σ.*

Інтерпретація h називається **моделлю** *для заданної множини формул Γ, якщо кожна формула із Γ істинна при інтерпретації h.*

Визначення 166. *Формула A називається* **логічно загальнозначимою**, *якщо вона істинна при довільній інтерпретації.*

Формула A називається **суперечністю**, *якщо $\neg A$ логічно загальнозначима, або те саме, що A хибна при довільній інтерпретації.*

Говорять, що формула A **логічний наслідок** *множини формул Γ, якщо при довільній інтерпретації формула A виконується на кожній послідовності цієї інтерпретації, на якій виконуються всі формули із Γ. Формули A і B називаються логічно еквівалентними, якщо кожна з них є логічним наслідком другої.*

Розглянемо такі прості твердження, які безпосередньо випливають з наведених вище означень.

Теорема 179. *Для довільних формул A і B мають місце такі властивості:*

(1) A хибна при даній інтерпретації тоді і тільки тоді, коли $\neg A$ істинна в тій самій інтерпретації, і навпаки;

(2) Ніяка формула не може бути одночасно хибною і істинною при одній і тій самій інтерпретації;

(3) Якщо в даній інтерпретації істинні формули A і $A \to B$, то в цій інтерпретації істинна і формула B;

(4) $A \to B$ хибна при даній інтерпретації тоді і тільки тоді, коли A істинна при цій інтерпретації, а B хибна;

(5) $A \wedge B$ виконана на послідовності s деякої інтерпретації тоді і тільки тоді, коли A виконана на s і B виконана на s.

$A \vee B$ виконана на послідовності s тоді і тільки тоді, коли або A виконана на s, або B виконана на s.

$A \leftrightarrow B$ виконана на s тоді і тільки тоді, коли або A виконана на s і B виконана на s, або коли A не виконана на s і B не виконана на s.

$\exists x_i A$ виконана на s тоді і тільки тоді, коли A виконана хоча б на одній послідовності s', яка відрізняється від послідовності s не більш ніж своїм i-м компонентом;

(6) A істинна при даній інтерпретації тоді і тільки тоді, коли $\forall x_i A$ істинна при даній інтерпретації;

(7) Кожний окремий випадок довільної тавтології істинний при довільній інтерпретації. (Окремим випадком тавтології називається формула, отримана шляхом підстановки формул у дану тавтологію замість пропозиційних літер, за тієї умови, що замість однієї і тієї самої літери підставляється одна і та сама формула);

(8) Нехай вільні змінні формули A, якщо вони існують, знаходяться серед змінних $x_i, x_j, \ldots, x_k$. Тоді якщо послідовності s і s' деякої інтерпретації мають однакові компоненти з номерами $i, j, \ldots, k$, то формула A виконана на послідовності s тоді і тільки тоді, коли вона виконана на послідовності s';

(9) Якщо формула A замкнута, то при довільній заданій інтерпретації або істинна A, або істинна $\neg A$;

(10) Нехай t і v — терми, s — деяка послідовність із Σ, t' отриманий із t шляхом підстановки v замість усіх входжень x_i, і s' отримана із s шляхом заміни в ній її i-ї компоненти на $s^*(v)$, тоді $s^*(t') = (s')^*(t)$;

(11) Якщо формула A не містить вільної змінної x_i, то формула $\forall x_i(A \to B) \to (A \to \forall x_i B)$ істинна при довільній інтерпретації.

Доведення пропонуються читачеві як вправи.

8.3. Аксіоматична система числення ППП

Теорія предикатів першого порядку **K** визначається аксіоматичним шляхом так.

Аксіоми цієї теорії поділяються на два класи: *логічні аксіоми* і *власні аксіоми*.

Логічні аксіоми: для довільних формул A, B, $C \in$ **K**, наступні формули є схемами аксіом теорії **K**:

(П1) $A \to (B \to A)$;
(П2) $(A \to (B \to C)) \to ((A \to B) \to (A \to C)$;
(П3) $(\neg B \to \neg A) \to ((\neg B \to A) \to B)$;
(П4) $\forall x_i A(x_i) \to A(t)$, де $A(x_i)$ формула із **K** і терм t із **K**, вільний для x_i в $A(x_i)$;
(П5) $\forall x_i(A \to B) \to (A \to \forall x_i B)$, якщо формула A не включає вільних входжень x_i.

Власні аксіоми не можуть бути сформульовані в загальному вигляді, оскільки вони змінюються від теорії до теорії.

Правила виведення:
(i) **Modus ponens:** із A і $A \to B$ випливає B.
(ii) **Правило узагальнення:** із A випливає $\forall x_i A$.

Визначення 167. *Теорія предикатів першого порядку з пустою множиною власних аксіом називається* **численням предикатів першого порядку (ЧППП),** *а теорія предикатів першого порядку з непустою множиною власних аксіом називається* **теорією першого порядку.**

Обмеження, наведені у формулюваннях аксіом, вимагають деяких пояснень. Якби в аксіомі П4) терм t міг би бути довільним, то стала б можливою така ситуація. Нехай $A(x) = \neg\forall y A(x, y)$ і $t = y$. Зауважимо, що t не є вільним для x в $A(x)$. Розглянемо такий випадок аксіоми П4:

$$\forall x(\neg\forall y\, A_1^2(x, y)) \to \neg\forall y\, A_1^2(y, y),$$

і візьмемо за область інтерпретації довільну область, що має не менше двох елементів, а A_1^2 інтерпретується як відношення тотожності. Тоді посилка істинна, а висновок хибний, отже, і вся формула хибна.

Аналогічна ситуація може мати місце і для аксіоми П5. Дійсно, нехай A і B являють собою $A_1^1(x)$. Таким чином, x входить вільно в A. Розглянемо окремий випадок аксіоми П5:

$$\forall x(A_1^1(x) \to A_1^1(x)) \to (A_1^1(x) \to \forall x\, A_1^1(x)).$$

Очевидно, що посилка в даній формулі істинна. Якщо вибрати інтерпретацію так, щоб $A_1^1(x)$ виконувалася не на всіх елементах області, а лише на деяких, то виявиться, що висновок у даній формулі буде хибним. Отже, і сама формула в даній інтерпретації також буде хибною.

Визначення 168. *Моделлю числення* **K** *називається довільна інтерпретація, при якій істинні всі аксіоми теорії* **K.**

Легко помітити (див. теорему 179), що коли правила виведення застосовувати до істинних у даній інтерпретації формул, то результатом є формули, які також істинні в тій самій інтерпретації. Отже, довільна **теорема**, тобто формула, яка виводиться в численні **K** із аксіом, істинна у довільній моделі числення **K**.

Теорія предикатів першого порядку з непустою множиною власних аксіом служить для задання теорій і їх моделей для різноманітних алгебр. Як приклад такої теорії розглянемо теорію напівгруп.

Приклад теорії першого порядку

Нехай **K** має один бінарний предикатний символ і один бінарний функціональний символ, які будемо записувати $t = s$ і $t + s$.

Власними аксіомами теорії **K** є формули:

а) $\forall x_1 \forall x_2 \forall x_3 (x_1 + (x_2 + x_3)) = ((x_1 + x_2) + x_3)$ (асоціативність);

б) $\forall x_1 (x_1 = x_1)$ (рефлексивність);

в) $\forall x_1 \forall x_2 (x_1 = x_2 \to x_2 = x_1)$ (симетричність);

г) $\forall x_1 \forall x_2 \forall x_3 (x_1 = x_2 \to (x_2 = x_3 \to x_1 = x_3)$ (транзитивність);

д) $\forall x_1 \forall x_2 \forall x_3 \, (x_2 = x_3 \to (x_1 + x_2 = x_1 + x_3) \wedge (x_2 + x_1 = x_3 + x_1)$ (підстановка);

Довільна модель цієї теорії називається **напівгрупою.** ♠

8.4. Основні властивості ЧППП і теорій першого порядку

Розглянемо основні властивості числення предикатів першого порядку. Насамперед доведемо такий факт.

Теорема 180. *Числення предикатів першого порядку* **K** *є несуперечною теорією.*

Доведення. Для довільної формули A із **K** нехай $h(A)$ означає формулу, яка отримана із формули A шляхом опускання в ній усіх кванторів і термів разом з усіма відповідними дужками і комами. Наприклад,

$$h(\forall x_2 A_3^2 (x_1, x_2) \to A_1^1 (x_1)) = A_3^2 \to A_1^1 .$$

$$h(\neg \forall x_5 A_2^3 (x_3, a_1, x_5) \to A_3^1 (x_4)) = \neg A_2^3 \to A_3^1 .$$

З означення h випливає, що $h(\neg A) = \neg h(A)$ і $h(A \to B) = h(A) \to h(B)$. А звідси маємо, що довільна аксіома П1)—П5) перетворюється на тавтологію. Дійсно, це очевидно для П1)—П3), а для 4) маємо

$$h(\forall x_i A(x_i) \to A(t)) = A \to A,$$

що теж є тавтологією (правило РІ). Далі, аналогічно до попереднього

$$h(\forall x_i(A \to B) \to (A \to \forall x_i B)) = (A \to B) \to (A \to B).$$

Отже, якщо $h(A)$ — тавтологія, то і $h(\forall x_i A)$ — тавтологія, крім того, як ми впевнились, коли $h(A)$ і $h(A \to B)$ — тавтології, то і $h(B)$ — тавтологія.

Таким чином, якщо $A \in \mathbf{K}$, то $h(A)$ — тавтологія. Якби існувала така формула B, що $B \in \mathbf{K}$ і $\neg B \in \mathbf{K}$, то формули B і $\neg B$ одночасно повинні були б бути тавтологіями, що неможливо в силу того, що числення висловлювань є несуперечною теорією. ■

Теорема 181. *Якщо $A \in \mathbf{K}$ — тавтологія, то A є теоремою $\mathbf{K}$ і може бути виведена в $\mathbf{K}$ лише з використанням аксіом П1)—П3) і правила MP.*

Доведення. Нехай формула A отримана з деякої тавтології B шляхом підстановок. Згідно з теоремою 156 існує виведення A в численні висловлювань. Зробимо в цьому виведенні всі можливі підстановки за таким правилом:

(і) якщо яка-небудь пропозиційна літера входить у B, то на місця всіх її входжень у кожну з формул виведення підставляємо ту формулу теорії $\mathbf{K}$, яка підставлялася в B;

(іі) якщо дана пропозиційна літера не входить у B, то на місця всіх її входжень у формулах виведення підставляємо довільну (але одну і ту саму) формулу теорії $\mathbf{K}$. Отримана таким чином послідовність формул і буде виведенням формули A в теорії $\mathbf{K}$, причому виведенням, яке використовує лише аксіоми П1)—П3) і правило MP. ■

Визначення 169. *Нехай A є формулою деякої сукупності формул Γ і $B_1, B_2, \ldots, B_n$ — деяке виведення із Γ з обґрунтуванням кожного кроку в ньому. Будемо говорити, що B_i залежить від A в цьому виведенні, коли*

а) B_i є A і обґрунтуванням B_i є належність B_i до Γ або

б) B_i обґрунтовується як безпосередній наслідок за правилом MP, або Gen деяких попередніх у цьому виведенні формул, із яких хоча б одна залежить від A.

Приклад 8.4.5. Довести, що $A, \forall xA \to B \vdash \forall xB$.

(1) A — гіпотеза, (залежить від A)
(2) $\forall xA$ — (1), Gen, (залежить від A)
(3) $\forall xA \to B$ — гіпотеза, (залежить від A)
(4) B — MP із (2) і (3), (залежить від A і $\forall xA \to B$)
(5) $\forall xB$ — Gen із (4), (залежить від A і $\forall xA \to B$). ♠

Теорема 182. *Якщо формула B не залежить від формули A при виведенні $\Gamma, A \vdash B$, то $\Gamma \vdash B$.*

Доведення індукцією за довжиною виведення $B_1, B_2, \ldots, B_n = B$.

При $n = 1$ маємо $B_1 = B$ і B або належить до $\Gamma \cup \{A\}$, або є аксіомою. Але B не може збігатися з A в силу незалежності B від A. Отже, $\Gamma \vdash B$.

Нехай теорема справедлива для всіх формул B, довжина виведення яких менша за n. Якщо $B \in \Gamma$ або B — аксіома, то $\Gamma \vdash B$. Якщо ж B — безпосередній наслідок деяких формул із $B_1, B_2, \ldots, B_{n-1}$ (однієї або двох), то оскільки B не залежить від A, то від A не залежить жодна з цих формул. Але тоді, за припущенням індукції, із Γ виводяться ці формули (одна або дві), а разом з ними і формула B. ■

Теорема дедукції для числення предикатів не може бути сформульована в такому вигляді, в якому вона формулювалася для числення висловлювань. Але деякий варіант цієї теореми існує.

Теорема 183 (*Теорема дедукції*). *Нехай $\Gamma, A \vdash B$ і водночас нехай існує таке виведення B із Γ, A, в якому при жодному застосуванні правила Gen до формул, які залежать від A в цьому виведенні, не зв'язується квантором загальності ніяка вільна змінна формули A. Тоді $\Gamma \vdash A \to B$.*

Доведення індукцією за довжиною виведення $B_1, B_2, \ldots, B_n = B$.

При $n = 1$ існує дві можливості:

а) $B_1 \in \Gamma$ або є аксіомою;

б) $B_1 = A$

У випадку а) $A \to B_1$, оскільки це випливає з аксіоми П1)

$$B_1 \to (A \to B_1)$$

за правилом МР.

У випадку б), якщо $A = B_1$, то $\Gamma \vdash A \to B_1$, оскільки $A \to A$ — тавтологія (правило РІ).

Нехай теорема справедлива для всіх $i < n$. Припустимо, що існують індекси k і j менші за i, такі що $B_k = B_j \to B_i$. Тоді, за припущенням індукції, $\Gamma \vdash A \to B_j$ і $\Gamma \vdash A \to (B_j \to B_i)$. Але за аксіомою П2 маємо

$$A \to (B_j \to B_i) \to ((A \to B_j) \to (A \to B_i))$$

і за правилом МР отримуємо, що $\Gamma \vdash A \to B_i$.

Припустимо, нарешті, що існує такий індекс $j < i$, що $B_i = \forall x_k B_j$. Тоді, за припущенням індукції, $\Gamma \vdash A \to B_i$ і або B_j не залежить від A, або x_k не є вільною змінною формули A. Якщо B_j не залежить від A,

то за теоремою 182, $\Gamma \vdash B_i$ і застосовуючи правило *Gen*, отримуємо $\Gamma \vdash \forall x_k B_i$. За схемою аксіом П1) знаходимо, що

$$B_i \to (A \to B_i)$$

і $A \to B_i$ — за правилом MP.

Якщо x_k не є вільною змінною формули A, то за схемою аксіом П5) маємо

$$\forall x_k(A \to B_j) \to (A \to \forall x_k B_j).$$

За припущенням індукції $\Gamma \vdash A \to B_i$, а за правилом *Gen* $\Gamma \vdash \forall x_k (A \to B_i)$ і, нарешті, за правилом MP отримуємо $\Gamma \vdash A \to \forall x_k B_i$, тобто $\Gamma \vdash A \to B_i$. ■

Наслідок 51. *Якщо $\Gamma, A \vdash B$ і існує виведення B без застосування правила Gen до вільних змінних формули A, то $\Gamma \vdash A \to B$.*

Наслідок 52. *Якщо формула A замкнута і $\Gamma, A \vdash B$, то $\Gamma \vdash A \to B$.*

Доведення обох наслідків випливають безпосередньо з теореми дедукції.

Приклад 8.4.6. Довести такі теореми:

1) $\vdash \forall x \forall y A \to \forall y \forall x A$.

a)	$\forall x \forall y A$	— гіпотеза;
b)	$\forall x \forall y A \to \forall y A$	— схема аксіом П4;
c)	$\forall y A$	— MP із *a*), *b*);
d)	$\forall y A \to A$	— схема аксіом П4;
e)	A	— MP із *c*), *d*);
f)	$\forall x A$	— Gen із *e*);
g)	$\forall y \forall x A$	— Gen із *f*).

Таким чином показано, що $\forall x \forall y A \vdash \forall y \forall x A$, причому в збудованому доведенні ні при якому застосуванні правила *Gen* не зв'язуються вільні змінні формули $\vdash \forall x \forall y A$. Отже, за наслідком 51 маємо $\vdash \forall x \forall y A \to \forall y \forall x A$.

Аналогічне обґрунтування має доведення наступної формули

2) $\vdash \forall x(A \to B) \to (\forall x A \to \forall x B)$.

a)	$\forall x(A \to B)$	— гіпотеза;
b)	$\forall x(A \to B) \to (A \to B)$	— схема аксіом П4;
c)	$A \to B$	— MP із *a*), *b*);
d)	$\forall x A$	— гіпотеза;
e)	$\forall x A \to A$	— схема аксіом П4;

f) A — MP із *d*), *e*);
g) B — MP із *c*), *f*);
h) $\forall xB$ — Gen із *g*);
i) $\forall x(A \to B) \to (A \to \forall xB)$ — схема аксіом П5;
k) $A \to \forall xB$ — MP із *a*), *i*);
l) $\forall xA \to A, A \to \forall xB) \vdash \forall xA \to \forall xB$ — ТІ.

3) $\vdash \forall x_1 \forall x_2 \ldots \forall x_n A \to A.$

a) $\forall x_1 \forall x_2 \ldots \forall x_n A$ — гіпотеза;
b) $\forall x_1 \forall x_2 \ldots \forall x_n A \to \forall x_2 \ldots \forall x_n A$ — схема аксіом П4;
c) $\forall x_2 \ldots \forall x_n A$ — MP із *a*), *b*);
d) $\forall x_2 \ldots \forall x_n A \to \forall x_3 \ldots \forall x_n A$ — схема аксіом П4;
... .
aa) $\forall x_n A$ — MP із ...;
bb) $\forall x_n A \to A$ — схема аксіом П4;
cc) A — MP із *aa*), *bb*).

Отже, формула A виводиться без застосування правила *Gen* і, за наслідком 51, остаточно отримуємо

$$\forall x_1 \forall x_2 \ldots \forall x_n A \vdash A$$

і

$$\vdash \forall x_i \forall x_2 \ldots \forall x_n A \to A. \spadesuit$$

На прикладі числення висловлювань ми бачили, наскільки корисним є розширення множини правил виведення. Такі правила дають можливість скоротити довжину виведення. Розглянемо деякі з таких правил для числення предикатів.

Правило індивідуалізації (RI). Якщо терм t вільний для змінної x у формулі $A(x)$, то $\forall xA(x) \vdash A(t)$.

Доведення.
a) $\forall xA(x)$ — гіпотеза;
b) $\forall xA(x) \to A(t)$ — схема аксіом П4;
c) $A(t)$ — MP із *a*), *b*).
Отже, $xA(x) \vdash A(t)$. ■

Правило існування (RE). Якщо терм t вільний для змінної x у формулі $A(x)$, то $A(t) \vdash \exists xA(x)$.

Доведення.
a) $\forall x \neg A(x) \to \neg A(t)$ — схема аксіом П4;
b) $A(t) \to \neg \forall x \neg A(x)$ — пр. контрапозиції із *a*).
Отже, $A(t) \to \exists xA(x)$, або $\vdash A(t) \to \exists xA(x)$. ■

Правило кон'юнкції (RK). Для довільних формул A і B має місце: $A, B \vdash A \wedge B$.

Доведення.

a)	A, B	— гіпотези;
b)	$A \to (B \to \neg(A \to \neg B))$	— тавтологія (див. ЧВ);
c)	$B \to \neg(A \to \neg B)$	— MP із *a*), *b*);
d)	$\neg(A \to \neg B) \leftrightarrow A \wedge B$	— MP із *a*), *c*). ■

Правило диз'юнкції (RD). Для довільних формул A і B має місце: $A \to C, B \to D, \neg A \to B \vdash \neg C \to D$.

Доведення.

a)	$A \to C, B \to D, \neg A \to B, \neg C$	— гіпотези;
b)	$(A \to C) \to (\neg C \to \neg A)$	— тавтологія;
c)	$\neg C \to \neg A$	— MP із *a*), *b*);
d)	$\neg C \to \neg A, \neg A \to B \vdash \neg C \to B$	— TI;
e)	B	— MP із *a*), *d*);
f)	$B \to D$	— гіпотеза;
g)	D	— MP із *e*), *f*). ■

Теорема 184. *Якщо A і B — формули теорії першого порядку* **K** *і x — предметна змінна, яка не є вільною змінною у формулі A, то справедливі такі теореми в* **K**:

(*i*1) $A \leftrightarrow \forall x A$;
(*i*2) $\exists x A \leftrightarrow A$;
(*i*3) $\forall x(A \to B) \leftrightarrow (A \to \forall x B)$;
(*i*4) $\forall x(B \to A) \leftrightarrow (\exists x B \to A)$;
(*i*5) $\forall x(A \leftrightarrow B) \to (\forall x A \leftrightarrow \forall x B)$.

Доведення.

(*i*1) a) A — гіпотеза;
b) $\forall x A$ — *Gen* із *a*).

Оскільки x не є вільною змінною в A, то $A \vdash \forall x A$ за теоремою дедукції. А це означає, що $\vdash A \to \forall x A$.

Доведення в інший бік і твердження (i2)—(i5) пропонуються як вправи. ■

Нехай змінні x і y не збігаються між собою і формула $A(y)$ отримана з формули $A(x)$ підстановкою y замість усіх вільних входжень x, тоді формули $A(x)$ і $A(y)$ називаються **подібними**, якщо y вільна для x в $A(x)$ і $A(y)$ не має вільних входжень y. Іншими словами, формули $A(x)$ і $A(y)$ подібні тоді і тільки тоді, коли $A(y)$ має вільні входження y в точності в тих місцях, у яких $A(x)$ має вільні входження x.

Лема 10. *Якщо формули $A(x)$ і $A(y)$ подібні, то* $\vdash \forall x A(x) \leftrightarrow \forall y A(y)$.

Доведення. За схемою аксіом П4) маємо $\vdash \forall x A(x) \to A(y)$. З умови леми і правила *Gen* отримуємо $\vdash \forall y(\forall x(A(x) \to A(y))$. За схемою аксіом П5) маємо $\vdash \forall x A(x) \to \forall y A(y)$.

Аналогічно доводиться і формула $\vdash \forall y A(y) \to \forall x A(x)$. Скориставшись тавтологією

$$A \to (\neg\neg B \to \neg(A \to \neg B) \leftrightarrow A \to (B \to (A \wedge B))$$

(тавтологія *f* із ЧВ) і теореми 181, отримуємо $\forall x A(x) \leftrightarrow \forall y A(y)$. ■

Лема 11. *Якщо замкнута формула $\neg A$ теорії першого порядку* **K** *не виводиться в теорії* **K**, *то теорія* **K′**, *отримана з* **K** *в результаті приєднання формули A як аксіоми, є несуперечною теорією.*

Доведення від супротивного. Нехай **K′** — суперечна. Це означає, що існує формула B, для якої має місце $\vdash B$ і $\vdash \neg B$ в теорії **K′**. Скориставшись тавтологією $\neg A \to (A \to B)$ (теорема *e*) із ЧВ) і теореми 181, маємо $\vdash B \to (\neg B \to \neg A)$ в теорії **K′**. Отже, у **K′** істинно $\vdash \neg A$ або, що те саме, що в теорії **K** A $\vdash \neg A$. Оскільки A — замкнута, то в силу наслідку 52 отримуємо $\vdash A \to \neg A$ в **K**. Але тоді із тавтології $(A \to \to \neg A) \to \neg A$ (теорема b) із ЧВ) отримуємо, що $\vdash \neg A$ в теорії **K**. Але останнє суперечить умові леми. ■

Лема 12. *Множина всіх виразів теорії першого порядку* **K** — *зліченна.*

Доведення. Нехай u — деякий символ алфавіту теорії першого порядку. Покладемо у відповідність символу u непарне число $g(u)$ за таким правилом:

$$g(() = 3,\ g()) = 5, g(,) = 7, g(\neg) = 9, g(\to) = 11, g(x_k) = 5 + 8k,$$
$$g(a_k) = 7 + 8k, g(f_k^n) = 9 + 8(2^n 3^k), g(A_k^n) = 11 + 8(2^n 3^k).$$

Далі, виразу вигляду $u_0 u_1 \ldots u_r$ поставимо у відповідність число $2^{g(u_0)} 3^{g(u_1)} p_r^{g(ur)}$, де p_r — r-те просте число. За такої відповідності, очевидно, різні вирази отримують різні номери і таким чином можна перелічити всі вирази в порядку зростання чисел, які поставлені їм у відповідність. ■

Визначення 170. *Теорія першого порядку* **K** *називається* **повною**, *якщо для довільної замкнутої формули A теорії* **K** *має місце* $\vdash A$ *або* $\vdash \neg A$ *в* **K**.

Теорія першого порядку **K′**, яка має такі самі символи, що й теорія першого порядку **K**, називається **розширенням теорії K**, якщо довільна теорема теорії **K** є також теоремою теорії **K′**. Очевидно, що для доведення того, що теорія **K′** є розширенням теорії **K**, необхідно довести, що всі власні аксіоми теорії **K** є теоремами теорії **K′**.

Лема 13. *Якщо теорія першого порядку несуперечна, то існує несуперечне повне її розширення.*

Доведення. Нехай $B_1, B_2,...$ — деякий перелік усіх замкнутих формул теорії першого порядку **K** (він існує в силу леми 12). Визначимо таку послідовність теорій $K_0, K_1,...$. Нехай $K_0 = K$ і припустимо, що K_n визначена. Якщо неправильно, що $\vdash \neg B_{n+1}$ у K_n, то додаємо формулу B_{n+1} до K_n як нову аксіому і отримуємо теорію K_{n+1}. А якщо $\vdash \neg B_{n+1}$, то покладаємо $K_n = K_{n+1}$. Нехай тепер K' — теорія першого порядку, аксіомами якої є всі аксіоми теорій $K_0, K_1,..., K_n,...$. Очевидно, що K_{n+1} служить розширенням для K_n, у тому числі і для теорії $K = K_0$. Для доведення несуперечності теорії K' достатньо довести несуперечність кожної з теорій K_i, оскільки довільне виведення суперечності в K' використовує лише скінченне число аксіом і тому є виведенням суперечності в деякій теорії K_n.

Доведення виконується індукцією за числом n. Для $n = 0$ теорія $K_0 = K$ несуперечна за умовою. Нехай теорія K несуперечна для всіх $i < n + 1$. Тоді якщо $K_{n+1} = K_n$, то і K_{n+1} несуперечна. Якщо ж $K_{n+1} \neq K_n$, то існує формула $\neg B_{n+1}$, яка не виводиться в K_n. Але за лемою 11 теорія K_{n+1} також несуперечна. Отже, несуперечна й теорія **K′**. Покажемо повноту теорії **K′**.

Нехай A — довільна замкнута формула теорії **K′**. Зрозуміло, що $A = B_{j+1}$ для деякого $j \in N$. Згідно з означенням теорії K_j або $\vdash \neg B_{j+1}$ у теорії K_j або $\vdash B_{j+1}$ у теорії K_{j+1}, оскільки формула B_{j+1} додається як аксіома в K_{j+1}. Таким чином, маємо або $\vdash \neg B_{j+1}$, або $\vdash B_{j+1}$ у теорії K_{j+1}. ■

Теорема 185 (*Гедель*). *Довільна несуперечна теорія першого порядку має модель, область якої — зліченна множина.*

Доведення цієї теореми опускається, а ознайомитися з її доведенням можна у [28].

Теорія, яка має областю інтерпретації зліченну множину, називається теорією із зліченною моделлю.

Теорема 186. *Довільна логічно загальнозначима формула теорії першого порядку* **K** *є теоремою теорії* **K**.

Доведення. Достатньо розглянути лише замкнуті формули теорії **K**, оскільки довільна формула B логічно загальнозначима тоді і тільки тоді, коли логічно загальнозначиме її замикання і B виводиться в **K** тоді і тільки тоді, коли виводиться її замикання (див. теорему 179 пункт (6)).

Нехай A — логічно загальнозначима замкнута формула із **K**. Якщо A не виводиться в **K**, то побудуємо теорію **K′** шляхом додавання формули $\neg A$ як аксіоми. Теорія **K′** несуперечна в силу леми 11 і має модель M у силу теореми Геделя. Оскільки формула $\neg A$ є аксіомою в **K′**, то і вона істинна в M. Але в силу загальнозначимості A, вона теж істинна в M. Отже, формули A і $\neg A$ одночасно істинні в M, чого не може бути в силу теореми 179 пункт (2). Таким чином формула A має бути теоремою теорії **K**. ■

Теорема 187 (*Теорема про повноту*). *Теоремами числення предикатів першого порядку є ті і тільки ті формули, які логічно загальнозначимі.*

Доведення. Згідно з теоремою 179, пункт (7), аксіоми, які задаються схемами П1)—П3), логічно загальнозначимі. Відповідно до пункту (10) теореми 179 і наслідку 52, а також пункту (11) теореми 179, загальнозначимими є також аксіоми, що породжуються схемами П4) і П5). Відповідно до пунктів (3) і (4) теореми 179, правила виведення *MP* і *Gen* зберігають властивість формул бути загальнозначимими. Таким чином, довільна теорема числення предикатів першого порядку логічно загальнозначима.

Справедливість другої частини цієї теореми випливає з теореми 186. ■

Наслідок 53. (*а*) *Формула A істинна в кожній зліченній моделі теорії* **K** *тоді і тільки тоді, коли* $\vdash A$ *в* **K**. *Отже A істинна в кожній моделі теорії* **K** *тоді і тільки тоді, коли* $\vdash A$ *в* **K**;

(*б*) *якщо в кожній моделі теорії* **K** *формула B виконується на кожній послідовності, на якій виконуються всі формули деякої множини формул* Γ, *то* $\Gamma \vdash B$ *в* **K**;

(*в*) *якщо формула B теорії* **K** *є логічним наслідком даної множини формул* Γ, *то* $\Gamma \vdash B$ *в* **K**;

(*г*) *якщо формула B теорії* **K** *є логічним наслідком формули A тієї самої теорії, то* $A \vdash B$ *в* **K**.

Наслідок 54 (*Теорема Скулема-Левенгейма*). *Якщо теорія першого порядку* **K** *має яку-небудь модель, то вона має і зліченну модель.*

Дійсно, якщо **K** має модель, то **K** несуперечна (див. теорему 179 пункт (2)). В силу теореми Геделя вона має зліченну модель. ■

Теорема 188 (*Теорема еквівалентності*). *Якщо формула B є підформулою формули A і формула A' отримана з формули A заміною яких-небудь (можливо і ні одного) входжень B формулою C, і якщо довільна змінна формули B або формули C, що є одночасно зв'язаною змінною формули A, з'являється у списку $y_1, y_2, \ldots, y_k$, то* $\vdash \forall y_1 \forall y_2 \ldots \forall y_k (B \leftrightarrow C) \rightarrow (A \leftrightarrow A')$.

Доведення індукцією за числом зв'язок і кванторів в A. Насамперед зауважимо, що коли ні одне входження B насправді не замінюється, то A збігається з A', і формула, яку потрібно вивести, є окремим випадком тавтології $B \rightarrow (A \leftrightarrow A)$. Якщо B збігається з A і це єдине входження замінюється на C, то формула, яку потрібно вивести, виводиться із аксіоми П4) (див. приклади використання теореми дедукції). Отже, надалі можна вважати, що B є власною підформулою формули A і щонайменше одне входження B замінюється.

Припустимо, що теорема справедлива для довільної формули з меншим числом кванторів і логічних зв'язок, ніж у A.

Випадок 1. A — елементарна формула. Тоді B не може бути власною підформулою формули A.

Випадок 2. A має вигляд $\neg D$. Тоді нехай A' являє собою $\neg D'$. За припущенням індукції $\vdash \forall y_1 \forall y_2 \ldots \forall y_k (B \leftrightarrow C) \rightarrow (D \leftrightarrow D')$. Звідси за допомогою тавтології $A \leftrightarrow B \rightarrow (\neg A \leftrightarrow \neg B)$ отримуємо $\vdash \forall y_1 \forall y_2 \ldots \forall y_k (B \leftrightarrow C) \rightarrow (A \leftrightarrow A')$.

Випадок 3. A має вигляд $D \rightarrow E$. Нехай $A' = D' \rightarrow E'$. За припущенням індукції $\vdash \forall y_1 \forall y_2 \ldots \forall y_k (B \leftrightarrow C) \rightarrow (D \leftrightarrow D')$ і $\vdash \forall y_1 \forall y_2 \ldots \forall y_k$ $(\mathrm{B} \leftrightarrow C) \rightarrow (E \leftrightarrow E')$. Застосовуючи тавтологію $((\mathrm{A} \leftrightarrow B) \wedge (C \leftrightarrow D)) \rightarrow ((A \rightarrow C) \leftrightarrow (B \rightarrow D))$, отримуємо $\vdash \forall y_1 \forall y_2 \ldots \forall y_k (B \leftrightarrow C) \rightarrow$ $\rightarrow (A \leftrightarrow A')$.

Випадок 4. A має вигляд $\forall x D$. Тоді A' являє собою $\forall x D'$. За припущенням індукції $\vdash \forall y_1 \forall y_2 \ldots \forall y_k (B \leftrightarrow C) \rightarrow (D \leftrightarrow D')$. Змінна x не є вільною змінною формули $\vdash \forall y_1 \forall y_2 \ldots \forall y_k (B \leftrightarrow C)$, бо якщо допустити, що це так, то x входила б вільно в B або в C і, оскільки x зв'язана в A, то x входила б до переліку $y_1, y_2, \ldots, y_k$ і була б зв'язаною змінною в $\vdash \forall y_1 \forall y_2 \ldots \forall y_k (B \leftrightarrow C)$. А це суперечить умові. Застосовуючи тепер аксіому П5), отримуємо $\vdash \forall y_1 \forall y_2 \ldots \forall y_k (B \leftrightarrow C) \rightarrow$ $\rightarrow \forall x (D \leftrightarrow D')$. Відповідно до теореми 184 (пункт (i5)), маємо $\forall x (D \leftrightarrow D') \rightarrow (\forall x D \leftrightarrow \forall x D')$. Користуючись тепер транзитивністю імплікації, остаточно отримуємо $\vdash \forall y_1 \forall y_2 \ldots \forall y_k (B \leftrightarrow C) \rightarrow (\forall x D \leftrightarrow$ $\leftrightarrow \forall x D')$ або $\vdash \forall y_1 \forall y_2 \ldots \forall y_k (B \leftrightarrow C) \rightarrow (A \leftrightarrow A')$. ■

Теорема 189 (*Теорема про заміну*). *Нехай формули A, A', B і C задовольняють умовам теореми еквівалентності. Якщо* $\vdash B \leftrightarrow C$, *то* $\vdash A \leftrightarrow A'$, а якщо $\vdash B \leftrightarrow C$ і $\vdash A$, *то* $\vdash A'$.

Теорема є простим наслідком теореми еквівалентності.

Теорема 190 (*Теорема про перейменування зв'язаних змінних*). *Якщо* $\forall x B(x)$ *є підформулою формули A, формула $B(y)$ подібна до формули $B(x)$ і A' отримана з A заміною хоча б одного входження* $\forall x B(x)$ *в A на* $\forall y B(y)$, *то* $\vdash A \leftrightarrow A'$.

Доведення випливає з леми 10 і теореми про заміну.

8.5. Нормальні форми формул ЧППП

Процес пошуку доведень теорем у численні предикатів першого порядку спрощується, якщо він застосовується до деякої «стандартної» форми формул ЧППП. Розглянемо одну з таких форм.

Визначення 171. *Говорять, що формула F ЧППП знаходиться в* **попередній нормальній формі** *тоді і тільки тоді, коли вона має вигляд*:

$$(Q_1 x_1)\,(Q_2 x_2)\,\ldots\,(Q_n\, x_n)A,$$

де $(Q_i x_i)$ є або $(\exists x_i)$ або $(\forall x_i)$ і всі x_i різні для різних i, а формула A не має кванторів. Приставка $(Q_1 x_1)\,(Q_2 x_2)\,\ldots\,(Q_n x_n)$ називається **префіксом**, *а формула A —* **матрицею.**

Нагадаємо, що коли формула A залежить від вільної змінної x, то це буде записуватися як $A(x)$, у протилежному випадку — просто A.

Теорема 191. *Для довільних формул F, G, H теорії першого порядку справедливі такі логічні еквівалентності*:

а) $\neg\forall x F(x) \leftrightarrow \exists x(\neg F(x))$;
б) $\neg\exists x F(x) \leftrightarrow \forall x(\neg F(x))$;
в) $\forall x F(x) \vee G \leftrightarrow \forall x(F(x) \vee G)$; $\forall x F(x) \wedge G \leftrightarrow \forall x(F(x) \wedge G)$;
г) $\exists x F(x) \vee G \leftrightarrow \exists x(F(x) \vee G)$; $\exists x F(x) \wedge G \leftrightarrow \exists x(F(x) \wedge G)$;
д) $\forall x F(x) \wedge \forall x H(x) \leftrightarrow \forall x(F(x) \wedge H(x))$;
е) $\exists x F(x) \vee \exists x H(x) \leftrightarrow \exists x(F(x) \vee H(x))$;
ж) $Q_1 x F(x) \wedge Q_2 x H(x) \leftrightarrow Q_1 x Q_2 y(F(x) \wedge H(y))$;
з) $Q_1 x F(x) \vee Q_2 x H(x) \leftrightarrow Q_1 x Q_2 y(F(x) \vee H(y))$.

Доведення. а) $\neg\forall x F(x) \leftrightarrow \exists x(\neg F(x))$. Нехай h — довільна інтерпретація на деякій області D. Якщо $\neg(\forall x F(x))$ істинна в інтерпрета-

ції h, то $\forall xF(x)$ хибна в h. Це означає, що існує такий елемент $s^*(x) \in D$, для якого $F(s^*(x))$ хибна, або те саме, що $\neg F(s^*(x))$ істинна в h. Отже, $\exists x(\neg F(x))$ істинна в h.

Далі, якщо $\neg(\forall xF(x))$ хибна в h, то $\forall xF(x)$ істинна в h. Це означає, що $F(x)$ виконується на всіх послідовностях із D, або що $\neg F(x)$ не виконується ні на одній такій послідовності із D. Отже, $\exists x(\neg F(x)$ хибна в h. Таким чином, $\neg(\forall xF(x)) \leftrightarrow \exists x(\neg F(x))$.

б) Доведення аналогічне доведенню а).

в) $\forall xF(x) \vee G \leftrightarrow \forall x(F(x) \vee G)$. Нехай маємо $\forall xF(x) \vee G \leftrightarrow$ $\leftrightarrow \neg(\forall xF(x)) \rightarrow G \leftrightarrow \exists x(\neg F(x)) \rightarrow G$ в силу а). Отже,

1) $\exists x(\neg F(x)) \rightarrow G$ — гіпотеза,
2) $\neg F(x) \rightarrow \exists x(\neg F(x))$ — правило *RE*,
3) $\neg F(x) \rightarrow G$ – ТІ із 1) і 2),
4) $\forall x(\neg F(x)) \rightarrow G) \leftrightarrow \forall x(F(x) \vee G)$ – *Gen* із 3).

Навпаки

1) $\forall x(F(x) \vee G) \leftrightarrow \forall x(\neg F(x) \rightarrow G)$ — гіпотеза,
2) $\neg F(x) \rightarrow G$ — правило *RI*,
3) $(\neg F(x) \rightarrow G) \rightarrow (\neg G \rightarrow F(x))$ — пр. контрап.,
4) $\neg G \rightarrow F(x)$ — MP із 2), 3),
5) $\forall x(\neg G \rightarrow F(x))$ — *Gen* і 4),
6) $\forall x(\neg G \rightarrow F(x)) \rightarrow (\neg G \rightarrow \forall xF(x))$ — схема аксіом П5),
7) $\neg G \rightarrow \forall xF(x)$ — *MP* із 5), 6),
8) $(\neg G \rightarrow \forall xF(x)) \rightarrow (\neg \forall xF(x) \rightarrow G)$ — пр. контрап.,
9) $\neg \forall xF(x)) \rightarrow G \leftrightarrow \forall xF(x) \vee G$ — MP із 7), 8).

Доведемо спочатку перший із законів г).

1) $\exists xF(x) \vee G \leftrightarrow \forall x(\neg F(x)) \rightarrow G$ — гіпотеза,
2) $\neg F(x) \rightarrow G$ — правило *RI*,
3) $(\neg F(x) \rightarrow G) \rightarrow \exists x(\neg F(x) \rightarrow G)$ — правило *RE*,
4) $\exists x(\neg F(x) \rightarrow G) \leftrightarrow \exists x(F(x) \vee G)$ — MP із 2), 3).

Навпаки

1) $\exists x(\neg F(x) \rightarrow G)$ — гіпотеза,
2) $\exists x(\neg F(x) \rightarrow G) \rightarrow (\neg F(x) \rightarrow G)$ — теорема 184 (*i*2),
3) $\neg F(x) \rightarrow G$ — MP із 1), 2),
4) $\forall x(\neg F(x)) \rightarrow \neg F(x)$ — схема аксіом П4),
5) $\forall x(\neg F(x)) \rightarrow G \leftrightarrow \neg \forall x(\neg F(x)) \vee G \leftrightarrow \exists x(F(x)) \vee G$ — ТІ.

Доведення інших законів із в) і г) тепер легко випливають із законів де Моргана. Дійсно,

$$\forall xF(x) \wedge G \leftrightarrow \neg\,(\neg \forall xF(x) \vee \neg G) \leftrightarrow \neg(\exists x(\neg F(x)) \vee \neg G) \leftrightarrow$$
$$\leftrightarrow \neg(\exists x(\neg F(x) \vee \neg G)) \leftrightarrow \neg(\exists x \neg(F(x) \wedge G) \leftrightarrow \forall x(F(x) \wedge G).$$

Аналогічно доводиться і другий закон із г).

Доведення решти еквівалентностей пропонуються як вправи. ■

З теореми 191 випливає такий метод побудови ПНФ.

Для того щоб привести формулу F до ПНФ використовують такі закони.

і) Для вилучення логічних зв'язок $\leftrightarrow$, $\rightarrow$:

а) $A \leftrightarrow B = (A \rightarrow B) \wedge (B \rightarrow A)$,

б) $A \rightarrow B = \neg A \vee B$.

іі) Для вилучення і внесення всередину формул знака заперечення:

в) $\neg((\forall x)F(x)) = (\exists x)(\neg F(x))$,

г) $\neg((\exists x)F(x)) = (\forall x)(\neg F(x))$,

д) $\neg\neg F = F$.

ііі) Для виносу кванторів:

1) $QxF(x) \vee G = Qx(F(x) \vee G)$,

2) $QxF(x) \wedge G = Qx(F(x) \wedge G)$,

3) $(\forall x)F(x) \wedge (\forall x)H(x) = \forall x(F(x) \wedge H(x))$,

4) $(\exists x)F(x) \vee (\exists x)H(x) = \exists x(F(x) \vee H(x))$,

5) $(Q_1x)F(x) \wedge (Q_2x)H(x) = (Q_1x)(Q_2z)(F(x) \wedge H(z))$,

6) $(Q_1x)F(x) \vee (Q_2x)H(x) = (Q_1x)(Q_2z)(F(x) \vee H(z))$.

Приклади 8.5.7. 1) Побудувати ПНФ для формули $F = (\forall x)P(x) \rightarrow$ $\rightarrow (\exists x)Q(x)$.

Розв'язок. $(\forall x)P(x) \rightarrow (\exists x)Q(x) = \neg((\forall x)P(x)) \vee ((\exists x)Q(x)) =$ $= (\exists x)\neg P(x) \vee (\exists x)Q(x) = (\exists x)(\neg P(x) \vee Q(x))$.

2) Побудувати ПНФ для формули $F = (\forall x)(\forall y)(((\exists z)P(x, z) \wedge$ $\wedge P(y, z)) \rightarrow (\exists u)Q(x, y, u))$.

Розв'язок. $(\forall x)(\forall y)(((\exists z)P(x, z) \wedge P(y, z)) \rightarrow (\exists u)Q(x, y, u)) =$ $= (\forall x)(\forall y)(\neg((\exists z)P(x, z) \wedge P(y, z)) \vee (\exists u)Q(x, y, u)) = (\forall x)(\forall y)((\forall z)$ $(\neg P(x, z) \vee \neg P(y, z)) \vee (\exists u)Q(x, y, u)) = (\forall x)(\forall y)(\forall z)(\exists u)(\neg P(x, z) \vee$ $\neg P(y, z)) \vee Q(x, y, u))$. ♠

Виходячи з наведених міркувань і законів перетворення формул у численні предикатів і)—ііі), можна запропонувати такий алгоритм побудови попередньої нормальної форми.

АЛГОРИТМ ПОБУДОВИ ПНФ (A)

Початок.

• *Крок* 1. Вилучаємо зв'язки $\leftrightarrow$, $\rightarrow$ за допомогою правил а), б).

• *Крок* 2. Переносимо знак $\neg$ всередину формул, користуючись правилами $\neg(\neg F) = F$, $\neg(F \vee G) = \neg F \wedge \neg G$, $\neg(F \wedge G) = \neg F \vee \neg G$ і правилами в), г) і д).

• *Крок* 3. Перейменовуємо зв'язані змінні, якщо є така необхідність.

• *Крок* 4. Використовуємо правила 1)—6) для отримання попередньої нормальної форми.

Кінець.

Обґрунтуванням даного алгоритму є твердження, яке безпосередньо випливає з теореми 191.

Теорема 192. *Алгоритм ПНФ перетворює довільну формулу A теорії* **K** *до такої формули B, яка знаходиться в попередній нормальній формі, що* $\vdash A \leftrightarrow B$ *в* **K**.

8.6. Скулемівські стандартні форми

Ми розглянули питання про зведення довільної формули числення предикатів першого порядку до попередньої нормальної форми. Оскільки матриця формули не включає кванторів, то її можна подати у кон'юнктивній нормальній формі.

Вважають, що формула P знаходиться в **кон'юнктивній нормальній формі (КНФ)**, якщо вона має вигляд $P = P_1 \wedge P_2 \wedge \ldots \wedge P_k$, де $P_i = A_{i1} \vee A_{i2} \vee \ldots \vee A_{il}$ і кожна A_{ij} — це атомарна формула, або заперечення атомарної формули. Нагадаємо процедуру побудови КНФ.

АЛГОРИТМ ПОБУДОВИ КНФ (*A*)

Початок.

• *Крок* 1. Вилучаємо зв'язки $\leftrightarrow$, $\rightarrow$ з формули A за допомогою правил а), б) (як в алгоритмі ПНФ).

• *Крок* 2. Вилучаємо подвійне заперечення за допомогою правила $\neg(\neg F) = F$, і переносимо знак $\neg$ до атомів, користуючись законами де Моргана (як в алгоритмі побудови ПНФ).

• *Крок* 3. Для отримання нормальної форми формули A використовуємо дистрибутивні закони:

$$F \vee (G \wedge H) = (F \vee G) \wedge (F \vee H);$$
$$F \wedge (G \vee H) = (F \wedge G) \vee (F \wedge H).$$

Кінець.

Приклад 8.6.8. Побудувати КНФ для формул: а) $(P \vee \neg Q) \rightarrow R$; б) $(P \wedge (Q \rightarrow R)) \rightarrow S$.

Розв'язок. а) $(P \vee \neg Q) \rightarrow R = \neg(P \vee \neg Q) \vee R = (\neg P \wedge \neg(\neg Q)) \vee R =$
$= (\neg P \wedge Q) \vee R = (\neg P \vee R) \wedge (Q \vee R)$;

б) $(P \wedge (Q \to R)) \to S = (P \wedge (\neg Q \vee R)) \to S = \neg(P \wedge (\neg Q \vee R)) \vee S =$
$= (\neg P \vee \neg(\neg Q \vee R)) \vee S = (\neg P \vee Q) \wedge (\neg P \vee \neg R)) \vee S = (\neg P \vee Q \vee S) \wedge$
$\wedge (\neg P \vee \neg R \vee S)$. ♠

Нехай F знаходиться в ПНФ $(Q_1x_1)\,(Q_2x_2)\,\ldots\,(Q_nx_n)\,A$, де A має КНФ. Нехай (Q_rx_r) — квантор існування в префіксі $(Q_1x_1)\,(Q_2x_2)\,\ldots\,(Q_nx_n)$, $r = 1, 2, \ldots, n$. Якщо ніякий квантор загальності не стоїть у префіксі лівіше (Q_rx_r), то вибираємо константу c, відмінну від інших констант, що входять до складу формули A, заміняємо всі x_r на c в A і викреслюємо (Q_rx_r) з префікса.

Якщо $(Q_{r_1}x_{r_1}), \ldots, (Q_{r_k}x_{r_k})$ — список усіх кванторів загальності, які зустрічаються лівіше $(Q_{r_k}x_{r_k})$ у префіксі, то вибираємо новий k-арний функціональний символ f, відмінний від інших функціональних символів, які входять до складу A, заміняємо всі входження x_r в A на $f(x_{r_1}, \ldots, x_{r_k})$ і викреслюємо $(Q_{r_k}x_{r_k})$ із префікса.

Потім застосовуємо цей самий процес до всіх кванторів існування в новому префіксі: остання із отриманих формул є **скулемівською стандартною формою.** Константи і функції, які при цьому використовуються для заміни змінних кванторів існування, називаються **скулемівськими функціями.** Нагадаємо деякі означення, які були наведені на початку даного розділу.

Диз'юнкт, який складається із k літер, називається k-літерним диз'юнктом. Однолітерний диз'юнкт називається **одиничним диз'юнктом.** Коли диз'юнкт не має ніяких літер, то його називають **пустим** диз'юнктом. Оскільки пустий диз'юнкт не має ніяких літер, які могли б виконуватись при довільній інтерпретації, то пустий диз'юнкт завжди хибний. Пустий диз'юнкт будемо позначати 0.

Вважається, що множина диз'юнктів S являє собою кон'юнкцію всіх диз'юнктів із S, де кожна змінна в S керована квантором загальності. Завдяки цій домовленості, скулемівська стандартна форма може бути представлена множиною диз'юнктів.

Теорема 193 (*Основна властивість скулемівських стандартних форм*). *Нехай S — множина диз'юнктів, які представляють стандартну скулемівську форму формули F. Тоді F хибна в тому і тільки в тому випадку, коли хибна S.*

Доведення. Не обмежуючи загальності, будемо вважати, що формула F знаходиться в попередній нормальній формі, тобто $F = Q_1x_1Q_2x_2\ \ldots\ Q_nx_nM(x_1, x_2, \ldots, x_n)$. Нехай Q_r — перший квантор існування і $F_1 = \forall x_1 \ldots \forall x_{r-1} Q_{r+1}x_{r+1}\ \ldots\ Q_nx_nM(x_1, x_2, \ldots, x_{r-1}, f(x_1, x_2, \ldots,$

x_{r-1}), x_{r+1}, …, x_n), де f — скулемівська функція, яка відповідає x_r, $r = 1, 2, \ldots, n$.

Покажемо, що F_1 суперечна тоді і тільки тоді, коли суперечна F. Припустимо, що F суперечна. Якщо F_1 несуперечна, то існує така інтерпретація h, що F_1 істинна в h, тобто для всіх $x_1, \ldots, x_{r-1}$ існує хоча б один елемент вигляду $f(x_1, \ldots, x_{r-1})$, для якого $Q_{r+1}x_{r+1} \ldots Q_n x_n M(x_1, x_2, \ldots, x_{r-1}, f(x_1, x_2, \ldots, x_{r-1}), x_{r+1}, \ldots, x_n)$, істинна в h. Таким чином, формула F істинна в h, а це суперечить припущенню. Отже, F_1 має бути суперечністю.

Припустимо тепер, що F_1 суперечна. Якщо F несуперечна, то існує така інтерпретація h в області D, що F істинна в h, тобто для всіх $x_1, \ldots, x_{r-1}$ існує такий елемент x_r, що формула $Q_{r+1}x_{r+1} \ldots Q_n x_n M(x_1, x_2, \ldots, x_{r-1}, x_r, x_{r+1}, \ldots, x_n)$, істинна в h. Розширимо інтерпретацію h шляхом введення функції $f(x_1, \ldots, x_{r-1})$, яка відображає $(x_1, \ldots, x_{r-1})$ в x_r для всіх $x_1, \ldots, x_{r-1}$ у D, тобто $f(x_1, \ldots, x_{r-1}) = x_r$. Позначимо нову інтерпретацію через h'. Зрозуміло, що для всіх $x_1, \ldots, x_{r-1}Q_{r+1}x_{r+1} \ldots Q_n x_n M(x_1, \ldots x_{r-1}, f(x_1, \ldots, x_{r-1}), x_{r+1}, \ldots, x_n)$ істинна в h', тобто F_1 істинна в h', що суперечить припущенню про суперечність F_1. Отже F має бути суперечністю.

Нехай F має m кванторів існування, $F_0 = F$ і формула F_k отримана із формули F_{k-1} заміною першого квантора існування в F_{k-1} скулемівською функцією, $k = 1, 2, \ldots, m$. Очевидно, що $S = F_m$. Із доведеного вище випливає, що F_k суперечна тоді і тільки тоді, коли суперечна формула F_{k-1} для $k = 1, 2, \ldots, m$. Отже, S суперечна тоді і тільки тоді, коли суперечна F. ■

Слід зауважити, що одна й та сама формула може мати більше ніж одну стандартну форму. Для простоти, при перетворенні формули F до скулемівської стандартної форми S, при виконанні замін кванторів існування скулемівськими функціями, останні вибираються настільки простими, наскільки це можливо.

8.7. Контрольні питання, задачі і вправи

1. Дайте означення логічно загальнозначимої формули.
2. Чим відрізняються формули числення висловлювань від формул числення предикатів?
3. Скільки аксіом і правил виведення має теорія першого порядку?
4. Скільки аксіом і правил виведення має числення предикатів першого порядку?
5. Сформулюйте основні властивості теорій першого порядку?

6. Сформулюйте теорему дедукції для числення висловлювань і числення предикатів.

7. Що таке стандартна форма формул числення предикатів першого порядку?

8. Побудувати доведення формули $(A \to B) \leftrightarrow (\neg B \to \neg A)$, не користуючись теоремою дедукції.

9. Знайти для формули $\neg(A \to (C \leftrightarrow B)) \to D$ еквівалентну їй формулу, яка включає лише логічні зв'язки $\vee$ і $\neg$, $\wedge$ і $\neg$.

10. Переконатися в тому, що аксіоми числення висловлювань і аксіоми числення предикатів є тотожно істинними формулами.

11. Нехай $P(x)$ означає «x — просте число», $E(x)$ — «x — парне число», $O(x)$ — «x — непарне число», $D(x, y)$ — «y ділиться на x». Перекладіть на українську мову такі формули числення предикатів першого порядку:

(а) $P(7)$;
(б) $E(2) \wedge P(2)$;
(в) $\forall x(E(x) \wedge D(x, 6))$;
(г) $\forall x(\neg E(x) \to \neg D(2, x))$;
(д) $\forall x(E(x) \wedge \forall y(D(x, y) \to E(y)))$;
(е) $\forall x(P(x) \to (\exists y)(E(y) \wedge D(x, y)))$;
(ж) $\forall x(O(x) \to (\forall y)(P(y) \to \neg D(x, y)))$;
(з) $(\exists x)\ (E(x) \wedge P(x)) \wedge \neg(\exists x)\ ((E(x) \wedge P(x)) \wedge ((\exists y)\ (x \neq y \wedge \wedge E(x) \wedge P(y))))$.

12. Нижче наведено п'ять речень українською мовою, за якими йдуть стільки ж речень символічної мови предикатів першого порядку. Знайдіть відповідні пари речень, так щоб кожний член пари був перекладом відповідного її члена.

(а) Усі судді ($J(x)$) — юристи ($L(x)$);
(б) Деякі юристи — шахраї ($S(x)$);
(в) Не всі юристи — судді;
(г) Жоден суддя не є шахраєм;
(д) Деякі юристи — політики ($P(x)$).

(а) $\exists x(L(x) \wedge S(x))$;
(б) $\exists x(L(x) \wedge P(x))$;
(в) $\neg(\forall x)(L(x) \to J(x))$;
(г) $\forall x(J(x) \to L(x))$;
(д) $\forall x(L(x) \to S(x))$.

13. Вказати вільні і зв'язані входження змінних у таких формулах:

a) $\forall z(\forall x A(x, y) \to B(z, x))$,
b) $\forall y A(z, y) \to \forall z A(z, y))$,
c) $(\forall y \exists y\ (A(x, y, f(x, y))) \vee \neg \forall x B(y, f(x))$.

14. Перекласти на мову формул такі речення:

а) Не всі птахи можуть літати.

б) Або кожний любить кого-небудь, і ніхто не любить усіх, або хтось любить усіх, і хтось не любить нікого.

в) Якщо хтось може це зробити, то і я теж можу це зробити.

г) Не всі люди щирі і не всі щирі люди багаті.

15. Чи є вільним терм $f(x, y)$ для x у формулах

1) $A(x, y) \to \forall x B(x)$,

2) $(\forall y A(y, a)) \vee \exists y A(x, y)$.

16. Чи будуть тотожно істинними такі формули:

a) $\exists x P(x) \to \forall x P(x)$;

b) $\neg(\exists x P(x) \to \forall x P(x))$;

c) $\exists x \forall y Q(x, y) \to \forall y \exists x Q(x, y)$;

d) $\forall x \exists y Q(x, y) \to \exists y \forall x Q(x, y)$.

17. Довести, що формула $\exists x \forall y(F(x, y) \to (\neg F(y, z) \to (F(x, x) \leftrightarrow \leftrightarrow F(y, y))))$ виконується у довільній моделі, яка має не більше трьох елементів.

18. Довести, що формула $(\forall x \exists y P(x, y) \wedge (\forall x \forall y P(x, y) \to \neg P(y, x)) \wedge \wedge \forall x \forall y \forall z(P(x, y) \to (P(y, z) \to P(x, z))))$ виконується у деякій нескінченній моделі і хибна у всіх скінченних моделях.

19. Побудувати ПНФ для формул

a) $\forall x \exists y ((A(x) \to B(y, z)) \to \exists x \forall z(B(x, z) \wedge A(y)))$;

b) $(\forall x P(x) \to \forall y(\forall z Q(x, z) \to \forall u P(u))))$.

20. Побудувати КНФ для таких формул:

a) $((P \to Q) \vee (Q \to P))$,

b) $((P \to Q) \vee (P \to \neg Q))$,

c) $(P \to (Q \to (P \wedge Q)))$,

d) $((P \to Q) \to ((Q \to R) \to (P \to R)))$,

e) $((\neg P \to \neg Q) \to (Q \to P))$,

f) $(P \to (Q \to P))$,

ж) $P \vee \neg P$,

g) $((P \to Q) \to ((P \to (Q \to R)) \to (P \to R)))$,

h) $(P \vee Q) \to P$,

i) $P \to (P \vee Q)$,

k) $(P \vee P) \to P$,

m) $Q \to (P \vee Q)$,

n) $(\neg P \to (P \to Q))$.

21. Побудувати скулемівські стандартні форми для формул із вправи 16.

22. Нехай A — формула, яка побудована із атомарних формул і їх заперечення за допомогою $\wedge$, $\vee$ і кванторів $\forall$, $\exists$ за довільними змін-

ними. Нехай A^+ — результат заміни у формулі A $\wedge$ на $\vee$, $\vee$ на формули на їх заперечення. Довести, що $A^+ \leftrightarrow A$. $\wedge$, $\forall$ на $\exists$, $\exists$ на $\forall$, атомарних.

23. Нехай формула A' отримана з формули A заміною $\wedge$ на $\vee$, $\vee$ на $\wedge$, $\forall$ на $\exists$, $\exists$ на $\forall$. Довести, що коли

а) $\vdash A \to B$, то $\vdash B' \to A'$;

b) $\vdash A \leftrightarrow B$, то $\vdash B' \leftrightarrow A'$.

24. Побудувати доведення формул

а) $\exists x \exists y A(x, y) \to \exists y \exists x A(x, y)$;

b) $\exists x \forall y A(x, y) \to \forall y \exists x A(x, y)$.

25. Довести, що коли множина формул Г виконується, то ця множина несуперечна.

26. Довести, що множина формул Г суперечна тоді і тільки тоді, коли довільна формула ЧППП виводиться із Г.

27. Довести, що довільна несуперечна множина формул ЧППП виконується (*теорема про існування моделі*).

ЛІТЕРАТУРА

1. *Ахо А., Хопкрофт Дж., Ульман Дж.* Построение и анализ вычислительных алгоритмов. — М.: Мир, 1979. — 535 с.

2. *Биркгоф Г., Барти Т.* Современная прикладная алгебра. — М.: Мир, 1976. — 400 с.

3. *Ван дер Варден Б. Л.* Алгебра. — М.: Наука, 1976. — 648 с.

4. *Васильев Ю. Л., Ветухновский Ф. Я., Глаголев В. В. и др.* Дискретная математика и математические вопросы кибернетики (т. 1). — М.: Наука, 1974. — 311 с.

5. *Виленкин Н. Я.* Комбинаторика. — М.: Наука, 1969. — 161 с.

6. *Виленкин Н. Я.* Популярная комбинаторика. — М.: Наука, 1975. — 207 с.

7. *Вирт Н.* Систематическое программирование. — М.: Мир, 1978. — 183 с.

8. *Глушков В. М.* Введение в кибернетику. — К.: Изд-во АН УССР, 1964. — 324 с.

9. *Глушков В. М., Цейтлин Г. Е., Ющенко Е. Л.* Алгебра, языки, программирование. — К.: Наук. думка, 1985. — 327 с.

10. *Гудстейн Р. Л.* Рекурсивный математический анализ. — М.: Наука, 1970. — 472 с.

11. *Евстигнеев В. А., Касьянов В. Н.* Теория графов (Алгоритмы обработки деревьев). — Новосибирск: ВО «Наука», 1994. — 360 с.

12. *Ежов И. П., Скороход А. П., Ядренко М. И.* Элементы комбинаторики. — М.: Наука, 1975. — 79 с.

13. *Емеличев В. А., Мельников О. И. и др.* Лекции по теории графов. — М.: Наука, 1990. — 382 с.

14. *Калужнин Л. А.* Введение в общую алгебру. — М.: Наука, 1973. — 447 с.

15. *Капітонова Ю. В., Кривий С. Л., Летічевский О. А., Луцький Г. М., Печурін М. К.* Основи дискретної математики. — К.: Наукова думка, 2002. — 578 с.

16. *Клини С.* Введение в метаматематику. — М.: ИЛ, 1957. — 674 с.

17. *Кон П.* Универсальная алгебра. — М.: Мир, 1968. — 351 с.

18. *Котов В. Е., Сабельфельд В. К.* Теория схем программ. — М: Наука, 1991. — 247 с.

19. *Кук Д., Бейз Г.* Компьютерная математика. — М.: Мир, 1990. — 383 с.
20. *Куратовский К., Мостовский А.* Теория множеств. — М.: Мир, 1970. — 416 с.
21. *Курош А. Г.* Лекции по общей алгебре. — М.: Наука, 1972. — 399 с.
22. *Курош А. Г.* Общая алгебра. Лекции 1969—70 гг. — М.: Наука, 1972. — 159 с.
23. *Лавров И. А., Максимова Л. Л.* Задачи по теории множеств, математической логике и теории алгоритмов. — М.: Наука, 1975. — 239 с.
24. *Линдон Р.* Заметки по логике. — М.: Мир, 1968. — 128 с.
25. *Мальцев А. И.* Алгебраические системы. — М.: Наука, 1970. — 370 с.
26. *Мальцев А. И.* Алгоритмы и рекурсивные функции. — М.: Наука, 1986. — 367 с.
27. *Марков А. А., Нагорный Н. М.* Теория алгорифмов. — М.: ФАЗИСТ, 1996. — 448 с.
28. *Мендельсон Э.* Введение в математическую логику. — М.: Мир, 1988. — 320 с.
29. *Новиков П. С.* Элементы математической логики. — М.: Наука, 1973. — 399 с.
30. *Оре О.* Теория графов. — М.: Наука, 1980. — 336 с.
31. *Пост Е.* Конечные комбинаторные процессы, формулировка 1. — В кн. «Машины Поста» / Под ред. В. А. Успенского. — М.: Наука, 1979. — С. 89—95.
32. *Райзер Г.* Комбинаторная математика. — М.: Мир, 1966. — 154 с.
33. *Савельев Л. Я.* Комбинаторика и вероятность. Новосибирск. — М.: Наука, 1975. — 452 с.
34. *Скорняков Л. А.* Элементы теории структур. — М.: Наука, 1982. — 158 с.
35. Справочная книга по математической логике. Теория рекурсии, т. 3. — М.: Мир, 1983.
36. *Столл Р.* Множества, логика, аксиоматические теории. — М.: Просвещение, 1968. — 230 с.
37. *Тейз А., Грибомон П., Луи Ж. и др.* Логический подход к искусственному интеллекту. — М.: Мир, 1990. — 429 с.
38. *Тейз А., Грибомон П., Юлен А. и др.* Логический подход к искусственному интеллекту (От модальной логики к логике баз данных). — М.: Мир, 1998. — 494 с.
39. *Уилсон Р.* Введение в теорию графов. — М.: Мир, 1977. — 205 с.
40. *Феллер В.* Введение в теорию вероятностей, т. 1. — М.: Мир, 1964. — 376 с.
41. *Филд А., Харрисон П.* Функциональное программирование. — М.: Мир, 1993. — 638 с.
42. *Френкель А., Бар-Хиллел И.* Основания теории множеств. — М.: Мир, 1966. — 328 с.
43. *Холл М.* Комбинаторика. — М.: Мир, 1970. — 234 с.
44. *Чень Ч., Ли Р.* Математическая логика и автоматическое доказательство теорем. — М.: Наука, 1983. — 256 с.

45. *Ben-Ari M.* Mathematical Logic for Computer Science. Prentice Hall International (UK) Ltd. — 1993. — 305 p.

46. *Gabbay D.* Elementary Logic (A procedural perspective). Prentice Hall Europe. — 1998. — 359 p.

47. *Goldblatt R.* Logics of Time and Computation. Lecture Notes. — N7. Center for the Study of Language and Information. — 1987.

48. *Coudert O., Madre J.C., Berthet C.* Verifying temporal properties of sequential machines without building their state diagrams. — Computer-Aided Verification'90. American Mathematical Society. — 1990. — P. 75—84.

49. *Johnson D. S.* A Catalogue of Complexity Classes — in J. van Leeuwen., ed. «Handbook of Theoretical Computer Science», vol. A. — 1990: Elsevier. — PP. 67—161.

50. *Papadimitriou C. H.* Computational complexity. — Addison-Wesley. — 1994. — 532 p.

51. *Peterson G. L.* Myths about the mutual exclusion problem. — Information Processing Letters, — 1981. — V. 12. — N 3. — P. 115—116.

52. *Гери М. Р., Джонсон Д. С.* Вычислительные машины и трудновычислимые задачи. — М.: Мир. — 1982. — 416 с.

Навчальне видання

КРИВИЙ Сергій Лук'янович

КУРС ДИСКРЕТНОЇ МАТЕМАТИКИ

Навчальний посібник

Коректор *Л. Гордієнко*
Художник обкладинки *Т. Зябліцева*
Верстка *Т. Мальчевської*

Підп. до друку 21.12.06. Формат 60×84/16. Папір офсет. № 1.
Гарнітура Тип Таймс. Друк офсет. Ум. друк. арк. 25,11.
Обл.-вид. арк. 29,37. Наклад 1500 пр. Зам. № 06-099.

Книжкове видавництво Національного авіаційного університету
03680, м. Київ, просп. Космонавта Комарова, 1
Свідоцтво про внесення до Державного реєстру
суб'єктів видавничої справи (серія ДК, № 977 від 05.07.2002)
Тел. (044) 406-78-33. Тел./факс: (044) 406-71-33
E-mail: publish@nau.edu.ua

Друк ФОП Гринчук І. Є.
03151, м. Київ, Повітрофлотський пр-т, 94 а
Тел. (044) 490-98-88